

A M&S Process to Achieve Reusability and Interoperability

**Nathalie Harrison, Bruno Gilbert, Marc Lauzon[†], Alfred Jeffrey[†],
Claire Lalancette[†], Dr Richard Lestage[†] and André Morin**

Electro-optical Warfare Section
[†]Precision Weapon Section
Defence R&D Canada – Valcartier
2459, boul. Pie-XI Nord
Val-Bélair, Québec, G3J 1X5
CANADA

Email: nathalie.harrison@drdc-rddc.gc.ca

ABSTRACT

This paper presents a modelling and simulation process based on state-of-the-art software engineering concepts, tools and best practices. It aims at guiding modellers in the development of extensible, reusable and interoperable models to be used in integrated simulations. Although this process has a very broad reach, it is incubated in a project for weapon system engagement simulation in order to better focus its application. The first iteration of the process development, which is reported here, is aimed at assessing the validity of the proposed concept. It was observed that despite the availability of tools and guidelines, no successful and cost-effective simulation is possible without teamwork, communication, common infrastructure, agreement, education, constraints and integration.

1.0 INTRODUCTION

In the context of defence related research and development, Modelling and Simulation (M&S) is often used as a tool to obtain precise answers to very specific questions. Due to time, resource and expertise constraints, several models or simulations are commonly developed in an executable format, with few customisable elements, to satisfy very specific requirements. Consequently, a more or less significant model rework is required for even slightly different applications. Another common weakness is the lack of rigorous, common and enforced modelling method, which often produces non-reusable and non-interoperable models.

The actual quest for a global synthetic environment is a catalyst to increase the span of M&S benefit. In this new vision, the real world is represented as a virtual environment where autonomous entities, behaviours, terrain, environment and information interact dynamically. A specific simulation only observes a subset of the entire environment. This conceptual approach may lead to some solutions to the M&S reusability and interoperability problem. However, the demanding system integration needs to be supported by effective teamwork and large-scale software development methods applied to the M&S domain.

From this point of view, new requirements for successful and cost-effective M&S include reusability, extensibility, portability, modularity and interoperability. Recently, the software engineering domain has tackled these problems with success. Therefore, since M&S relies on software applications, these novel M&S requirements could be fulfilled using state-of-the-art software engineering concepts, tools and best practices.

Paper presented at the RTO NMSG Conference on “NATO-PfP/Industry/National Modelling and Simulation Partnerships”, held in Paris, France, 24-25 October 2002, and published in RTO-MP-094.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 00 NOV 2003		2. REPORT TYPE N/A		3. DATES COVERED -	
4. TITLE AND SUBTITLE A M&S Process to Achieve Reusability and Interoperability				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Electro-optical Warfare Section Precision Weapon Section Defence R&D Canada Valcartier 2459, boul. Pie-XI Nord Val-Bélair, Québec, G3J 1X5 CANADA				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited					
13. SUPPLEMENTARY NOTES See also ADM001655., The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 52	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

A M&S Process to Achieve Reusability and Interoperability

This paper proposes an integrated M&S process to guide the modellers in the development of reusable and interoperable models to be used inside extensible simulation frameworks. The proposed process has been initiated by modellers from the engineering and engagement levels, at the bottom of the M&S levels pyramid (Figure 1). The team adopted a vertical approach and focussed on the modelling process. Model engineering [1] is essential to create models for high-fidelity sub-system simulations as well as for a higher-level synthetic environment. Emphasize was then on potential uses for the models instead of specific deliverables, which implied that modellers had to understand the different contexts in which their models may be useful.

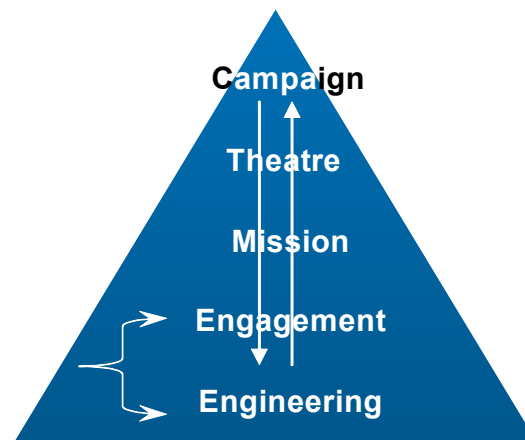


Figure 1: The M&S Levels of Detail Pyramid [2].

However, since modellers are rarely software engineers, an integration specialist is required. To foster the application of the proposed process, this person must understand the physics and the software aspects of the problem. It stresses the point that, to maximize reusability and interoperability, the bottom of the pyramid must understand the top and the top understand the bottom. For instance, the modellers must accept to be constrained, to some extent, to ensure that their models will be reusable and interoperable. The process must allow the modellers to focus on what they are good at, the modelling of physics.

This paper first describes the proposed M&S process and, afterwards, its practical implementation using specific software tools. The results section explains the incubating project surrounding the development of this process. Finally, the concluding remarks presents the lessons learned from the development and application of the proposed M&S process.

2.0 THE PROPOSED M&S PROCESS

As shown on Figure 2, the typical phases of any simulation are: modelling, execution and analysis [1]. The proposed process takes place to complement the M&S development theory in guiding the M&S teams in its concrete application. It essentially relies on software engineering concepts, tools and best practices applied to the M&S domain.

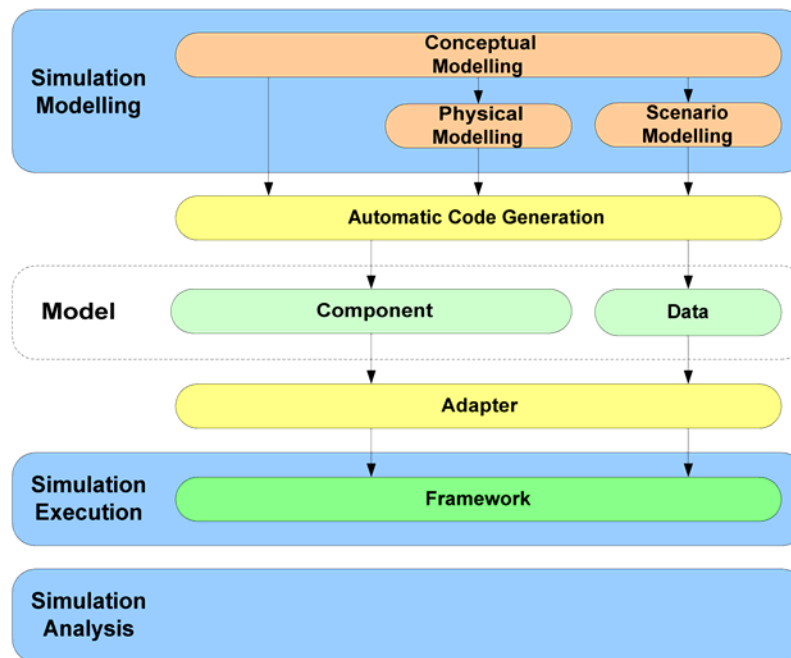


Figure 2: The Proposed M&S Process.

This is also an iterative process in the sense that the top-down integration can be restarted at any time in the development of a simulation and be completed within a few minutes time scale. In the software engineering domain, it is often called a micro-process spiral life cycle [3]. The automated steps standardize and speed up the model development process.

2.1 Simulation Modelling

The simulation modelling includes three main activities: 1) the conceptual modelling of the elements to be simulated; 2) the modelling of the physical behaviour of each element; and 3) the modelling of the scenarios describing the interaction between the various elements.

2.1.1 Conceptual Modelling

Conceptual modelling [1] is the first step of any structured M&S process. It is the abstraction of the entities, properties, behaviours and interactions to be simulated. In the proposed M&S process, the conceptual modelling is the reference of design. It means that modellers shall always go back at this level to make any changes. From a very high level perspective, representing the simulation requirements, it is possible to progress toward a model closer to the implementation.

In a software engineering approach to M&S, modularity and reusability can be expressed using the object-oriented (OO) paradigm and the component-oriented approach. As a standard for representing these concepts, the Unified Modelling Language (UML) [4] was selected to support the conceptual modelling. Therefore, applications of the simulation are described using “use case” diagrams while the static and dynamic aspects of the simulated system are represented using “class” and “sequence” diagrams, respectively. Finally, the implementation of the system is conceptualized using “component” diagrams. A consequence of this

A M&S Process to Achieve Reusability and Interoperability

approach is the fact that modellers must be educated on UML and object-oriented programming to agree on a common conceptual model.

Design patterns [5] and domain specific standards, such as the Real-Time Platform Reference Federation Object Model (RPR-FOM) [6] in defence M&S, are also introduced at this level to synthesize reproducible and globally accepted concepts.

A simulation developed with the proposed process gains its extensibility from the incremental additions at the conceptual model level. This process fosters reusability by inviting the modellers to extract the commonality of their models. It also promotes interoperability by forcing them to agree on standard concepts and interfaces.

2.1.2 Physical Modelling

Physically based modelling [7] is the mathematical representation of the real world behaviour. In order to remain at a manageable complexity level, the physical models are tailored to satisfy specific use cases. Various physical models may then be necessary for different applications, which is also called multi-modelling [1][8]. This concept is essential for bottom/up and top/down reusability of models in the M&S pyramid (Figure 1).

Once the stakeholders agree on requirements and the modellers agree on concepts (entities and interactions), each specialist can model the physics that is under its responsibility. Stand-alone work is possible at the condition of strictly respecting the conceptual model or updating it appropriately if a modification is required.

At the final stage of the physical modelling, the model is implemented into a software format. The software model can be directly written in an object-oriented programming language or encapsulated into a class if it is a legacy model. However, since modellers are not necessarily programmers, it is often impossible to require structured and standardized code from them. On the other hand, they can be assisted by automatic code generation tools providing a standardized code skeleton limiting the code to be written. Visual programming and simulation tools are also favoured for physical modelling without specific programming skills. Indeed, these tools were especially created for specific domain engineering-level rapid prototyping, test and validation. They often allow the reuse of functionalities through common libraries and the interoperability with other specialists. The only constraint is then to design physical models compliant with the OO and component-oriented conceptual model and to break the functionalities into discrete, more or less fine-grained, modules. The main challenge resides in switching from block and wire to object thinking and to manage the time steps synchronization between the integration schemes.

2.1.3 Scenario Modelling

The execution of a simulation requires the prior modelling of the scenario. In the OO approach, conceptual and physical modelling are dealing with generic objects while scenario modelling refers to instances of these objects. It typically includes the specification of model parameters, initial conditions for the state variables, the dynamic assembling of sub-models composing higher-level models, the recording of output results and the instantiation of objects composing a simulation scenario.

Scenario modelling generates the data characterizing the simulation. By scripting, in opposition to hard coding, these elements, it is possible to reuse parameters and initial conditions, to select the output to be logged and to dynamically compose, at run-time, the parts of a model or the entities of a scenario. A standard and flexible data format, such as the eXtensible Markup Language (XML) [9], can significantly foster the exchange, the modularity and the portability of these data.

2.2 Automatic Code Generation

To automate and speed up the M&S process, software tools can generate the code of the model components and data. This practice promotes software quality and model consistency. The model code generation can be done directly from the conceptual model. This option involves a manual intervention of the modeller to include, in the skeleton, his physical model written in the appropriate programming language.

Alternatively, the modeller can use a visual programming tool to develop his physical model and, afterwards, automatically generate the model code including all the behaviours. This option allows reusing the visual prototype to produce the final model components. Similarly, a tool associated to scenario modelling can automatically generate the model data.

2.3 Model

The outcome of the modelling phase is a software model that includes a component and its associated data. The component is the generic software implementation of a model while the data contains the features of each instance. For example, the simulation of two different instances of a model can be achieved by using the same model component with different data files.

To maximize the model modularity and reusability, the components must encapsulate fine-grained generic code modules. Moreover, in order to optimize the modeller's efforts, the components shall contain physical model code independent of any simulation framework. In practice, components are generally compiled in a library that can be dynamically instantiated and linked at run-time.

Model data specify what is left generic in the model components, each component having customizable parameters and initial conditions. Parameters are the model data that remain constant over the simulation while initial conditions change over time.

Entities are container models that can be dynamically composed of part models using configuration files. Similarly, simulation scenarios are composed of entity models with their respective configuration files.

2.4 Adapter

To maximize reusability, the generic models and data files are adapted (from the "Adapter" design pattern [4]) to specific simulation frameworks. The adapter relationship with the simulation framework and the model component is shown on Figure 3. An instance of the adapter is required for each instance of a model entity.

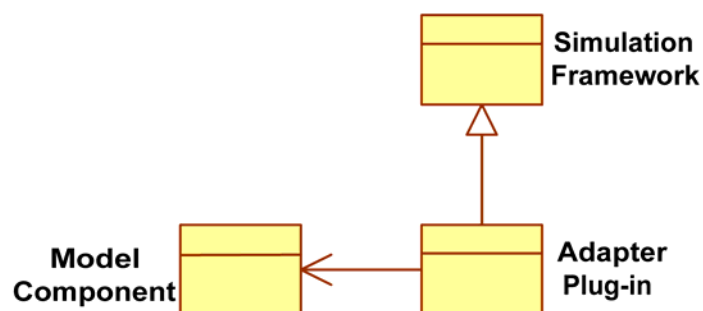


Figure 3: The "Adapter" Concept to Integrate a Model with a Framework.

A M&S Process to Achieve Reusability and Interoperability

Theoretically, the adaptation is possible with any OO and component-based simulation framework. However, a different adapter must be built to fit each specific framework. For a particular framework, the anchoring points can be limited to one or a few adapters depending on its extensibility and the compliance of the concepts between the custom models and the framework. Depending on the architecture and the complexity of the interaction, a variable level of effort may be required to benefit from the various built-in functionalities of a framework.

Practically, a model adapter is programmed using the Application Programming Interface (API) of the selected simulation framework and inserted into it as a plug-in. It acts as a dynamic model component interface with the simulation framework. The association of the model with the framework then does not require any recompilation and, depending on the framework, it may be run-time selectable. Scenario data file adapters can take the form of import/export capabilities.

Adapters have the advantage of reducing the dependence on a particular software product or environment. They contribute to improve the reusability of generic models, the modularity of the dependence to simulation execution and the extensibility of the simulation framework.

2.5 Simulation Execution

In the proposed process, the simulation execution is delegated to an existing commercial-of-the-shelf (COTS) framework that provides functionalities such as scenario creation, execution control, doctrines, trajectory, viewers, etc. Some frameworks may even include built-in model components and data. This approach originates from the fact that the expertise of the process initiators mainly resides in simulation modelling not in time management, distributed simulation, terrain database, visualization, etc.

The use of a recognized simulation framework contributes to the interoperability between the modellers by providing a common infrastructure. Within the defence community, such interoperability may also be improved if the chosen framework is compliant with the High-Level Architecture (HLA) [10].

2.6 Source Control and Web Page

The proposed M&S development process needs to be supported by version control, ownership tracking and exchange functionalities to ensure the integrity of the information. This can be achieved using, for instance, a shared database and a project web page. This practice shall be applied through the entire M&S process including: the conceptual model; the visual prototypes; the source code; the model components; the data files; and the documentation.

3.0 THE M&S PROCESS SUITE OF TOOLS

Practically, several software engineering tools are required to support and automate the proposed M&S process. An option analysis was carried out to determine the most appropriate suite of tools to demonstrate the proposed process. Some are COTS solutions and others are custom tools especially developed to be as generic as possible. It should be noted that these tools are not a unique solution but represent the best compromise when the option analysis was conducted. The automatic integration of these tools prevents subjective manual operations, promotes the iterative refinement and accelerates the process. Figure 4 shows the software tools associated to the M&S process while Figure 5 presents screen snapshots of the different tools.

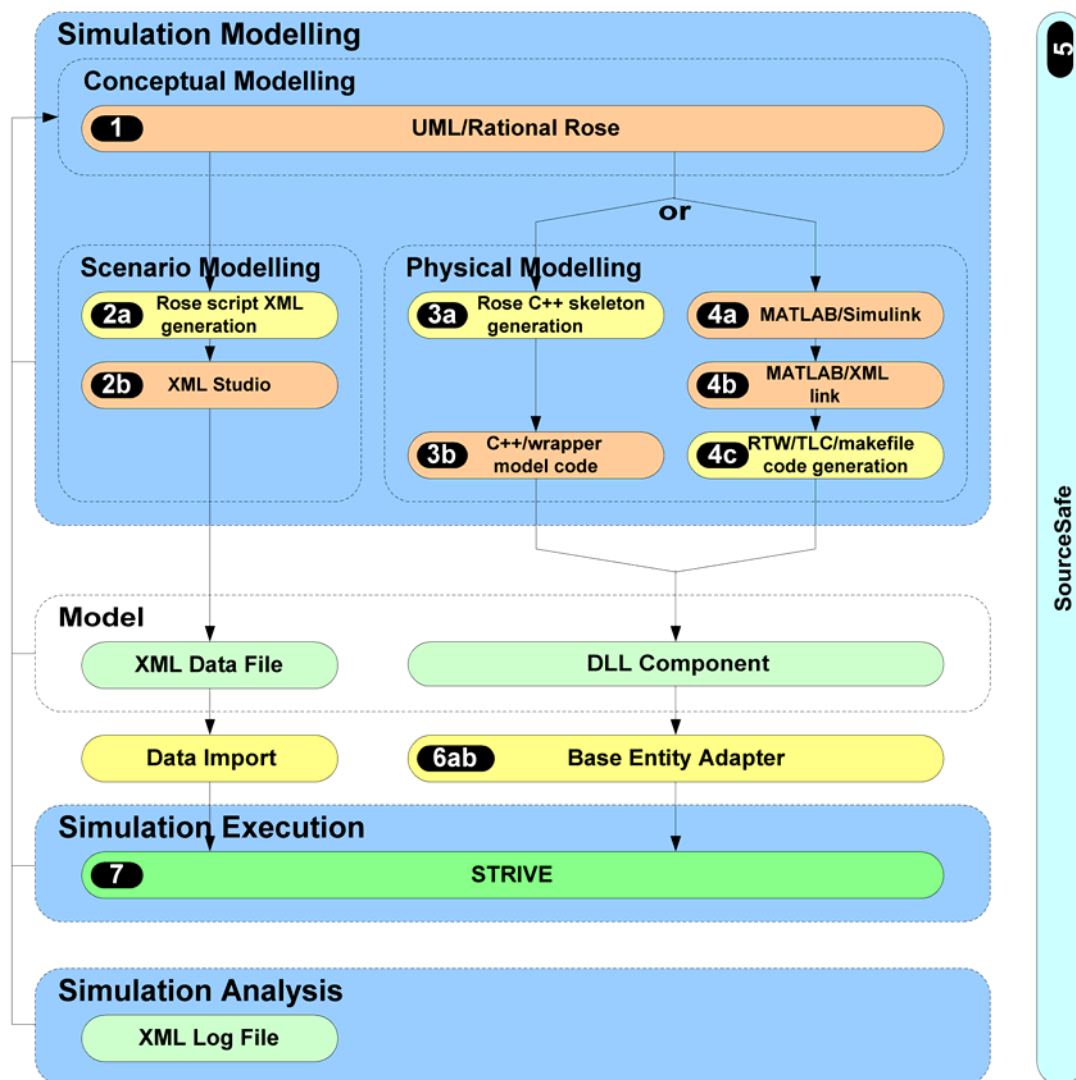


Figure 4: The Concrete Steps Implementing the Proposed M&S Process.

A M&S Process to Achieve Reusability and Interoperability

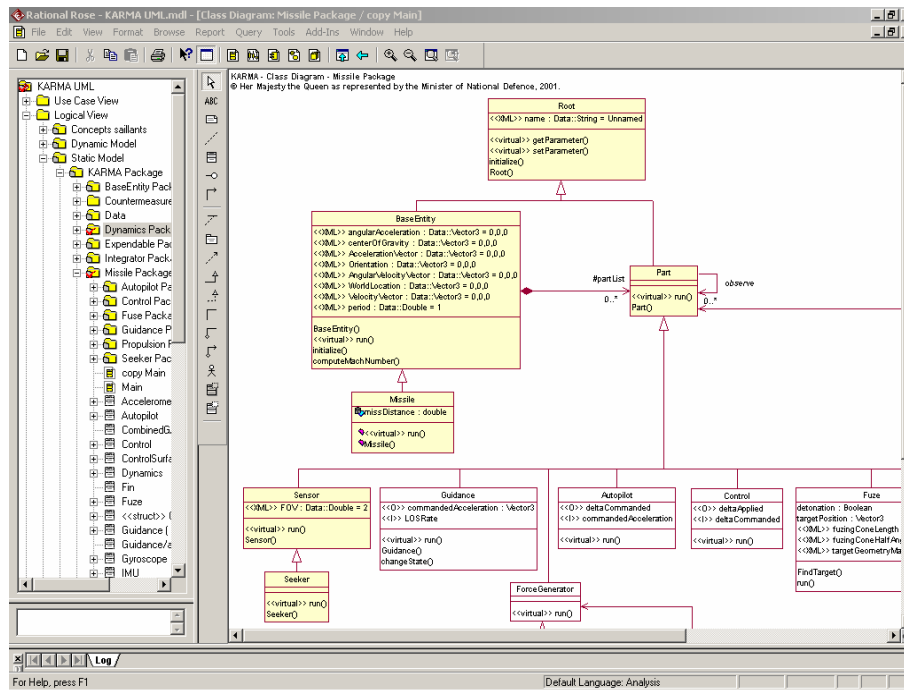


Figure 5a: Step 1 – Agreement on a Conceptual Model in UML with Rational Rose®.

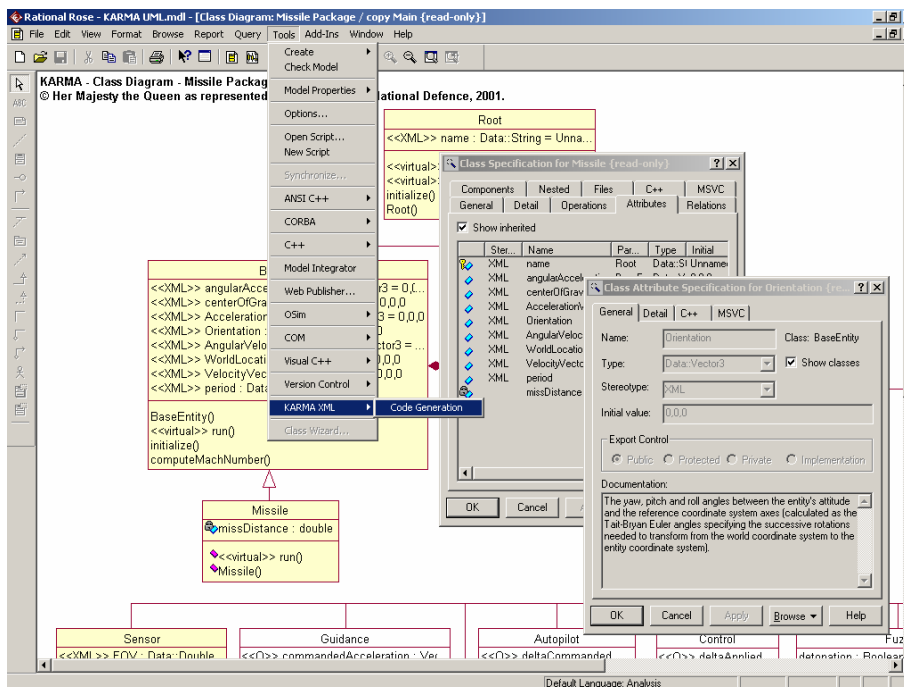


Figure 5b: Step 2a – Generation of XML Default Parameters Files from Rational Rose®.

Figure 5: Screen Snapshots of the Tools Supporting the M&S Process.

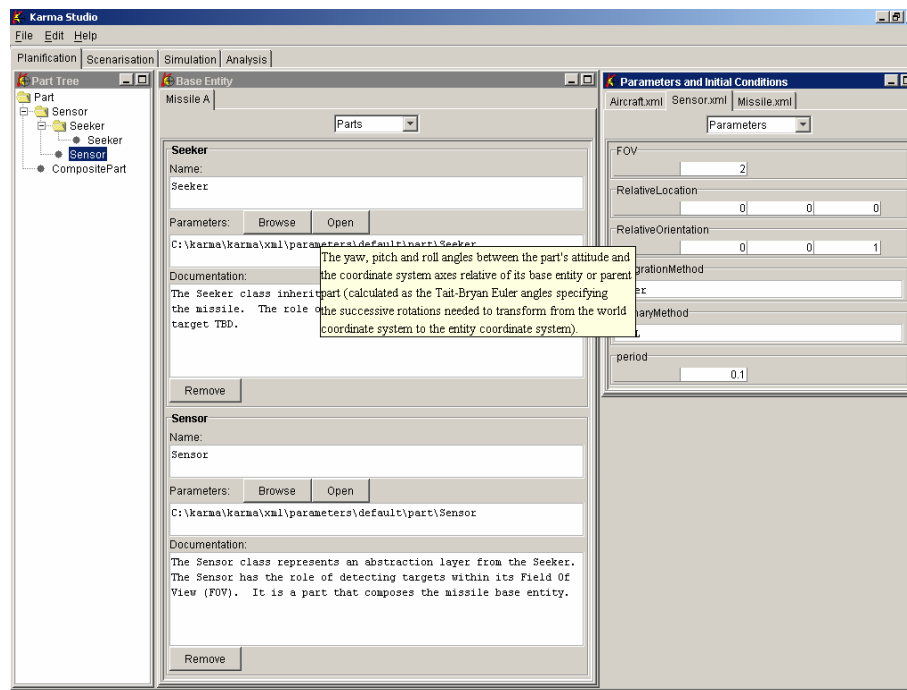


Figure 5c: Step 2b – Edition of the XML Scenario, Parts and Parameters Files in the Adaptive Java Studio.

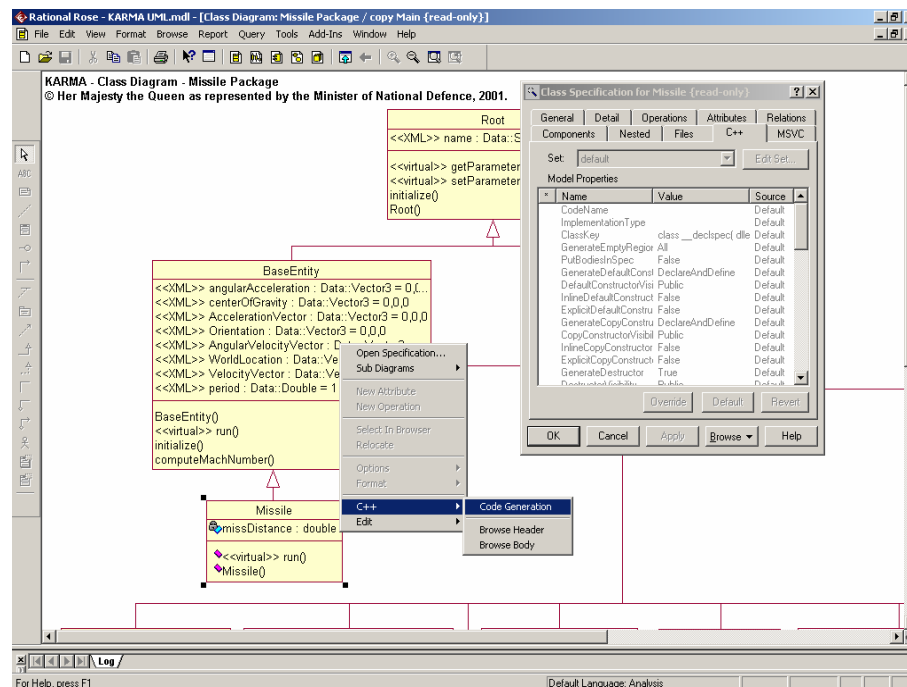


Figure 5d: Step 3a – Generation of a C++ Skeleton from the UML Conceptual Model with Rational Rose®.

Figure 5 cont'd: Screen Snapshots of the Tools Supporting the M&S Process.

A M&S Process to Achieve Reusability and Interoperability

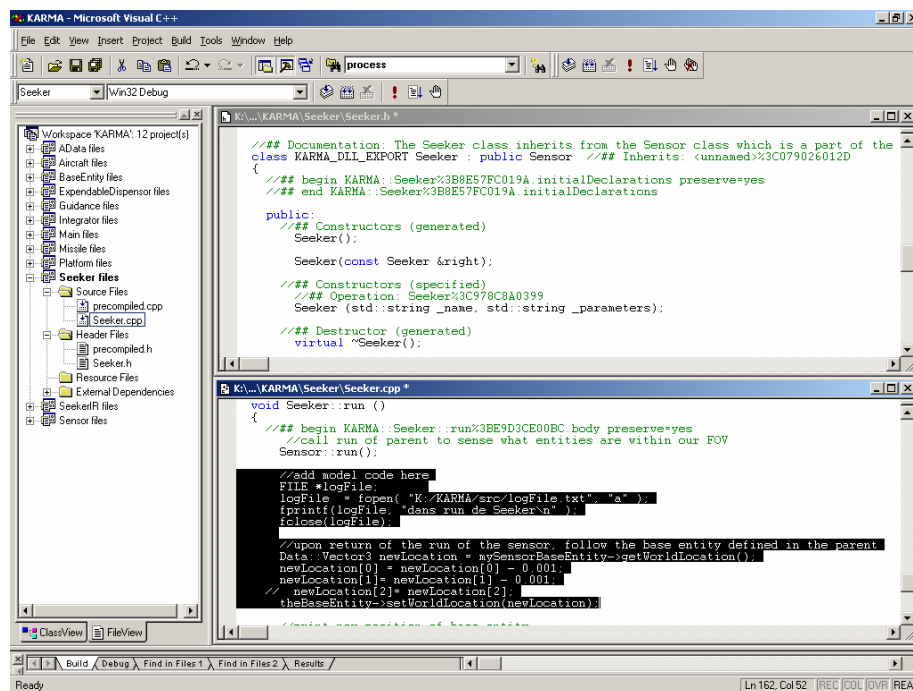


Figure 5e: Step 3b – Filling the C++ Skeleton with a Physical Model in C++ or Wrap a Legacy Model.

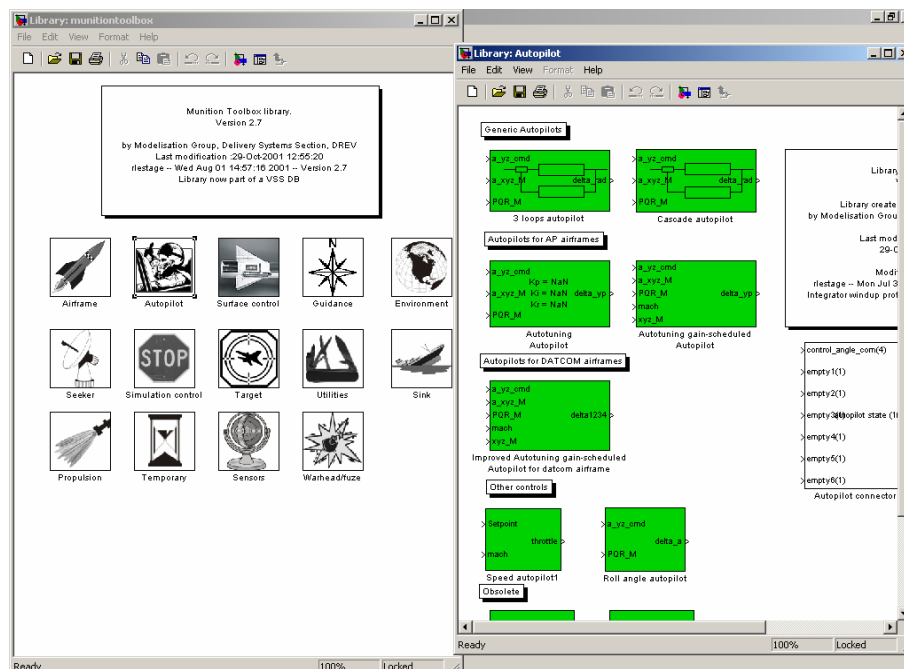


Figure 5f: Step 4a – Prototyping the Physical Model in MATLAB/SIMULINK®.

Figure 5 cont'd: Screen Snapshots of the Tools Supporting the M&S Process.

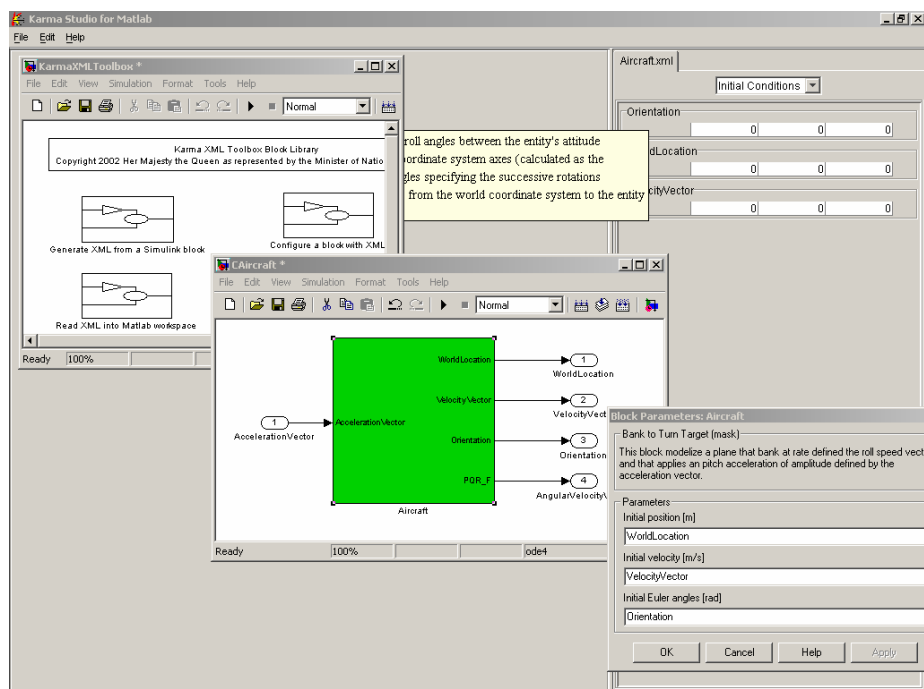


Figure 5g: Step 4b – Associating the SIMULINK® Model with XML Data Files Using the XML Toolbox.

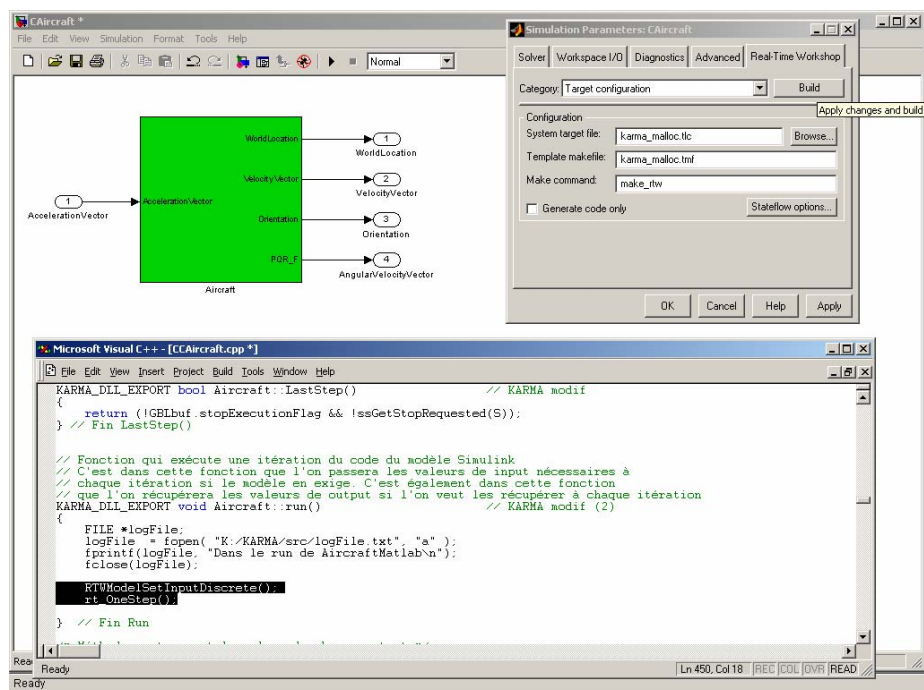


Figure 5h: Step 4c – Generating the DLL Using RTW®, TLC® Custom Script and Makefile.

Figure 5 cont'd: Screen Snapshots of the Tools Supporting the M&S Process.

A M&S Process to Achieve Reusability and Interoperability

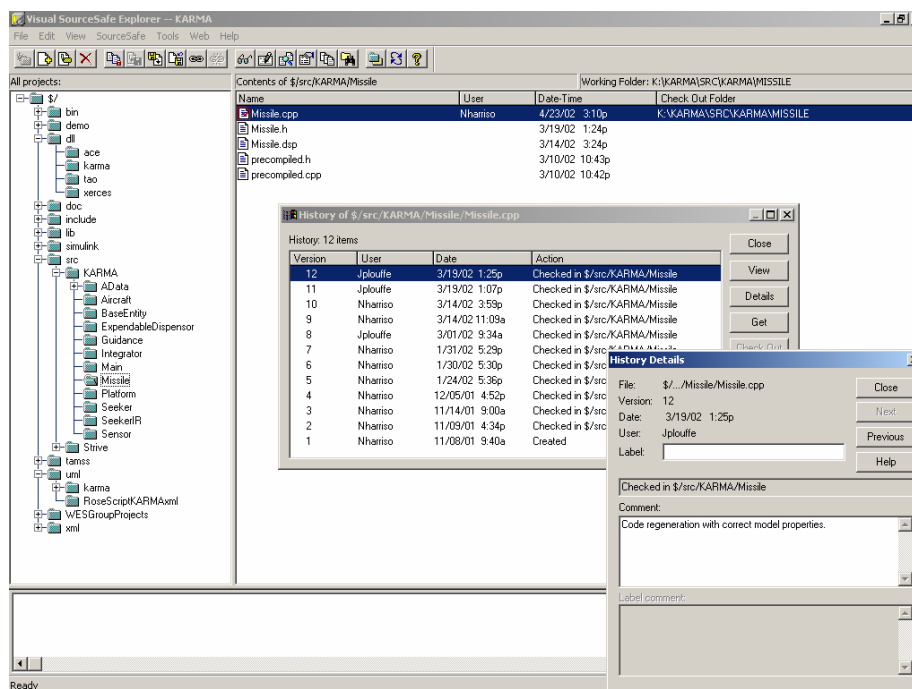


Figure 5i: Step 5 – Controlling the Versions and Sharing the Project Files with Sourcesafe®.

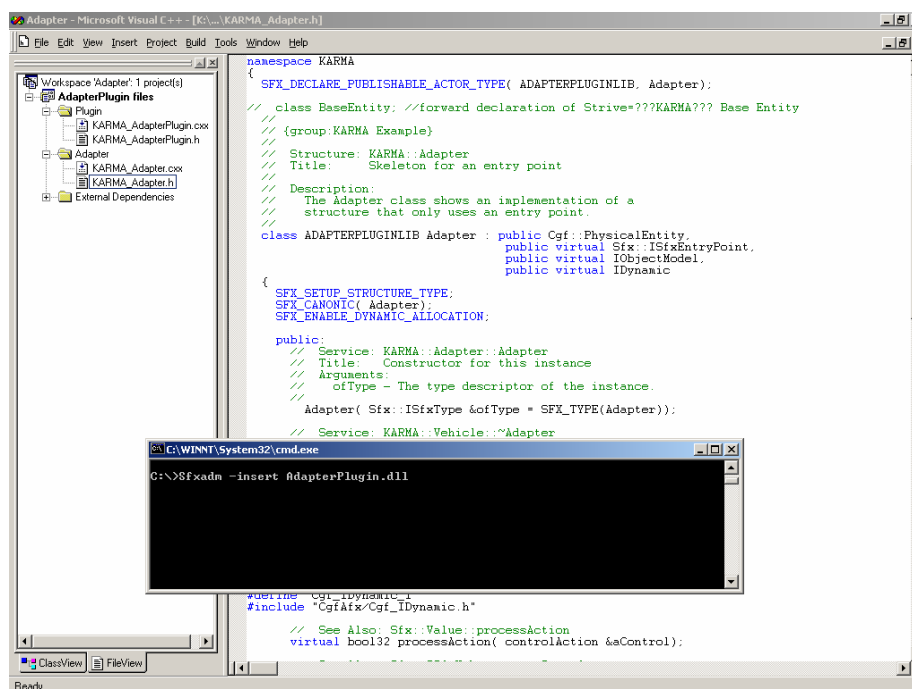


Figure 5j: Step 6a – Developing a STRIVE® Adapter in Microsoft Visual C++® and Inserting the Plug-In into the SFX.

Figure 5 cont'd: Screen Snapshots of the Tools Supporting the M&S Process.

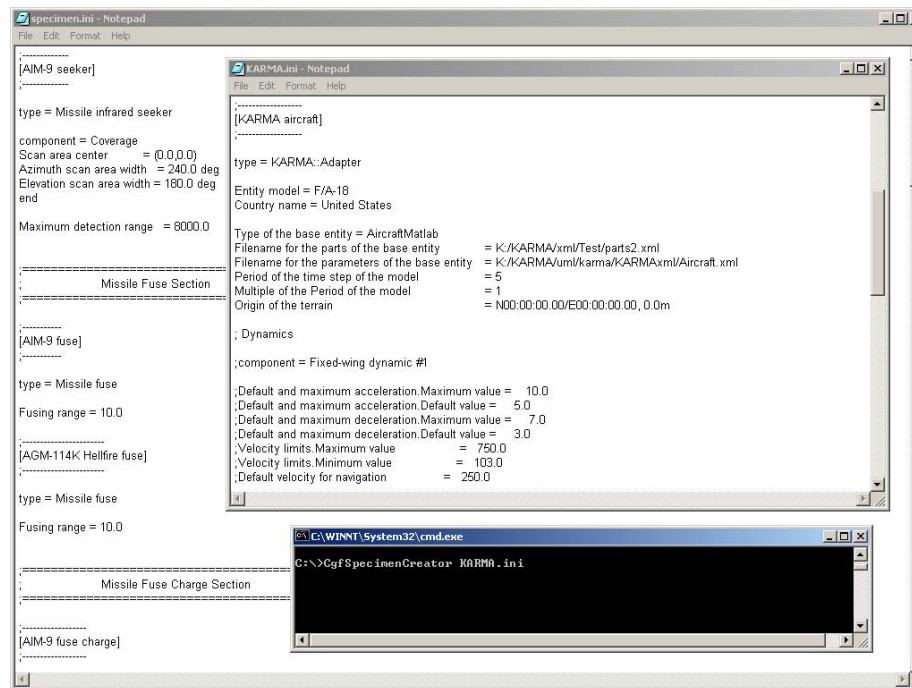


Figure 5k: Step 6b – Adapting the Data Files using a Custom STRIVE® specimen.ini File.

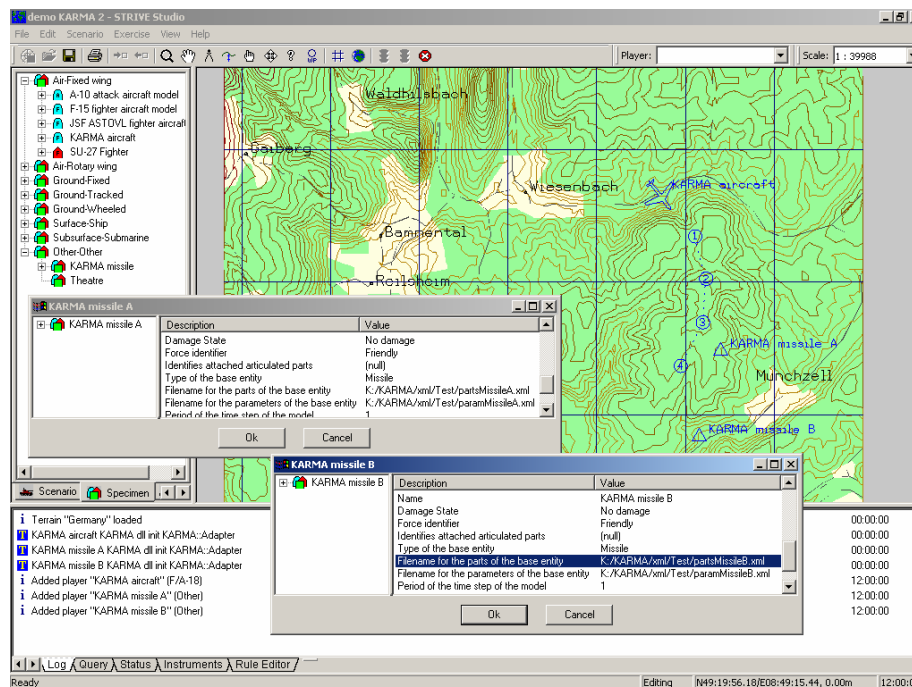


Figure 5l: Step 7 – Executing the Simulation in STRIVE®.

Figure 5 cont'd: Screen Snapshots of the Tools Supporting the M&S Process.

3.1 Simulation Modelling

Simulation modelling includes a series of tools to support the conceptual, scenario and physical modelling. Their link with each step of the proposed process is shown on Figure 4.

3.1.1 Conceptual Modelling

The process starts (Figure 4, Step 1 and Figure 5a) by agreeing on a conceptual model in UML using the Rational Rose[®] visual modelling tool [11]. It is a COTS product implementing the UML design standard and supporting OO and component-oriented conceptual modelling. The typical elements of the UML notation (use cases, class, sequence, component diagrams) are used to represent the conceptual model. The class attributes (parameters or initial conditions), methods, interaction and documentation are also systematically included in the UML class diagram.

3.1.2 Scenario Modelling

The second step of the proposed process (Figure 4, Step 2a and Figure 5b) consists in generating the XML parameter and initial condition files from the UML conceptual model using an automated functionality integrated into Rational Rose[®]. Practically, this is done in stereotyping, with the tag <<XML>>, the classes that require a parameter file. All class attributes that need to be configurable, either as a parameter or as an initial condition, are also tagged. Then, a custom script, integrated into the Rational Rose[®] tools menu, automatically generates default XML parameter files with the corresponding parameter names, data types, initial values and documentation. The practice of properly documenting the conceptual model and automatically generating the data insures the quality of the parameters data files.

Afterwards, the creation and the edition of the XML data files can also be done using a custom Java interface called the Studio (Figure 4, Step 2b and Figure 5c). This interface automatically adapts the Graphical User Interface (GUI) to the number and the name of the parameters defined into one model. It also dynamically represents the parameters according to their type. The data type entered by the user is validated and the minimum and maximum ranges permitted for each parameter are verified. Units and documentation are also displayed. The drag-and-drop capability of the GUI allows composing scenarios from entity models and entities from part models. The Xerces C++ and Java XML parsers [12] allow reusing the XML files at the scenario graphical modelling level, at the visual programming level and at the simulation framework level.

3.1.3 Physical Modelling

Based on the conceptual model, a C++ skeleton of the physical model can be automatically generated with Rational Rose[®] (Figure 4, Step 3a and Figure 5d). Moreover, Rational Rose[®] is integrated with the Microsoft Visual C++[®] development environment. The practice of systematically referring to the UML model to make any change on the skeleton and redo the automatic code generation ensures that the conceptual model always reflects the state of the implementation. It also improves the quality and the uniformity of the documentation and the code.

The modeller can then add the physical model directly in the reserved area of the C++ skeleton (Figure 4, Step 3b and Figure 5e) using Microsoft Visual C++[®] or any other appropriate development environment. At this stage, the modeller also has the possibility of wrapping a legacy model into the C++ skeleton of the class.

Alternatively, the modeller has the opportunity to use the MATLAB/SIMULINK® [13] visual programming tool for prototyping the physical model (Figure 4, Step 4a and Figure 5f). However, the SIMULINK® model should be consistent with the OO conceptual model to ensure the compliance between the SIMULINK® and the skeleton generated code.

When using MATLAB/SIMULINK®, the modeller shall associate the visual model with a XML data file (Figure 4, Step 4b and Figure 5g) to be consistent within the process. Custom tools were developed to use the XML data files with MATLAB/SIMULINK®: 1) XML files for SIMULINK® models are automatically generated using a m-file export program called MATLAB2XML; 2) parameters defined in XML can be imported in the MATLAB® workspace using the XML2MATLAB m-file program; and 3) the SIMULINK® blocks can be automatically associated with the XML Studio using an automatic configuration m-file.

If the modeller uses the MATLAB/SIMULINK® environment to develop the physical model, the proposed process allows to automatically generate a model component compiled as a Dynamic Link Library (DLL). The component is produced using the Real-Time Workshop® (RTW®) and the Target Language Compiler® (TLC®) COTS products (Figure 4, Step 4c and Figure 5h). RTW® is integrated within SIMULINK® and automatically generates portable and executable C code from the block model. Using the TLC®, included with RTW®, it is possible to customize the generated code and, for instance, wrap the produced C code into a C++ class compliant with the conceptual model. The interface of the resulting class is identical to the one automatically generated from the conceptual model with Rational Rose®. In addition, this code contains the calls to the MATLAB/SIMULINK® functions responsible for the mathematical modelling and the numerical integration. Finally, a custom makefile automatically compiles the generated code into a DLL to produce a stand-alone component.

3.2 Source Control

The modelling process produces model data for parameters, entity parts and scenarios in XML file format and model components in DLL file format. The different versions of these files are tracked using Microsoft Visual SourceSafe® COTS product (Figure 4, Step 5 and Figure 5i). SourceSafe® is integrated with the other modelling tools (Microsoft Visual C++®, Rational Rose® and MATLAB®) to optimize file management. Practically, it automates the sharing and the version control of the UML conceptual model, the C++ source code and the SIMULINK models.

3.3 Adapter

STRIVE® from CAE (Montreal, Canada) [14] is the simulation framework selected to demonstrate the process. It is an HLA-native framework that internally uses publish and subscribe as interaction mechanism. It implements distributed simulation using the Run-Time Infrastructure (RTI) [10] and supports an extended RPR-FOM. Its architecture is divided into two main elements: 1) a simulation framework, called SFX and 2) a Computer Generated Forces (CGF). It allows adding custom models that only use the SFX or use also the behaviours provided by the CGF.

Custom models are added into STRIVE® as plug-in components in DLL file format. To avoid coding the physical models using framework-dependent API, the proposed M&S process uses an adapter between the generic model DLLs and the STRIVE® SFX. The adapter can be initialized from a library template automatically installed by STRIVE® into Microsoft Visual C++® (Figure 4, Step 6a and Figure 5j). In a single command line, the plug-in is inserted into the STRIVE® framework.

A M&S Process to Achieve Reusability and Interoperability

Similarly, the XML data files shall be adapted through a STRIVE[®] specimen initialization file (Figure 4, Step 6b and Figure 5k). This step represents the minimal effort required so that custom models could be recognized within the STRIVE[®] CGF.

3.4 Simulation Execution

Finally, the simulation is executed in STRIVE[®] (Figure 4, Step 7 and Figure 5l) to benefit from its built-in functionalities i.e., HLA compliance, distributed capabilities, visual scenario and doctrine creation tools, trajectory waypoints, 2D and 3D viewers, etc. Nevertheless, the custom model remains responsible for initializing dynamically its parameters from the XML files and the simulation results are always logged into XML files to maximize their portability.

4.0 THE INCUBATING PROJECT FOR THE M&S PROCESS DEVELOPMENT

This M&S process has been demonstrated in the context of an R&D project aiming at developing a weapon system engagement simulation environment. Typical requirements for such engagement simulations are, for example, to simulate a specific threat X, with the parameters Y, engaging a target Z that could counteract in specific ways. Implemented using a classical approach, this would have resulted in narrow simulation capabilities. With the use of the proposed M&S process, it has been demonstrated that various configurable subparts developed by several specialists can be connected and interchanged dynamically in the STRIVE[®] simulation framework.

The conceptual modelling allowed to devise and properly structure the main concepts of the simulation i.e., autonomous “Base Entities”, composed of “Parts” equipments, are detecting each other within the “Environment” of the simulation “Theatre”. Standardized terms from the RPR-FOM (such as “BaseEntity”, “WorldLocation”, “VelocityVector”, etc.) were adopted to describe similar concepts. In order to meet the engagement simulation requirements, the base entity concept was specialized, for example, in “Aircraft” or “Weapon” and the part concept, in “Airframe”, “Sensor” or “Propulsion”. The conceptual model proved to be independent of the number and the assembly of instances.

The physical models of the parts used in the simulation were exported to DLL components from an existing MATLAB/SIMULINK[®] weapons library. All parameter, initial condition, base entity parts composition, scenario and log files were associated to XML files for universal use across the entire M&S process.

The execution of the simulation in STRIVE[®] allowed to play different scenarios by dynamically instantiating “Aircraft” instances composed of interchangeable parts, each with all configurable parameters. Through STRIVE[®], the models also became HLA compliant.

The main objective of the incubating project was to establish a solid architecture and good teamwork practices such as information sharing and documentation. The experimentation of the process showed that such methodology and tools greatly improve the quality of the end product while easing further developments.

5.0 CONCLUSION

This paper proposed an automated iterative process, supported by a suite of software engineering tools and best practices, to guide modellers in the development of reusable and interoperable models. The application of this process brought to light the following advantages:

- the reusability of component models independent of any simulation framework;
- the interoperability improvement from an agreement at the conceptual level;
- the modularity of the models XML data and DLL components;
- the extensibility of the conceptual model and the simulation framework;
- the portability of the simulation data in XML format; and
- the quality and consistency of the outcome due to many automated steps.

On the other hand, the application of the process also showed some noticeable disadvantages like:

- the maintenance of custom tools developed to support the process;
- the uncertainty of being at the mercy of COTS tools providers;
- the significant integration work requiring specific skills;
- the learning curve of the modellers for the conceptual modelling; and
- the rigorous information (database) management required.

Through the experience of the incubating project, the following lessons were learned:

- despite the availability of tools and guidelines, no successful and cost-effective M&S could be achieved without a major change of mind about teamwork in the defence R&D community;
- transparent and efficient information sharing and appropriate communication and documentation must be established within the team;
- reusability and interoperability only occurs with an agreement at the conceptual model level;
- modellers must be left to do what they are the best at, the mathematical modelling of physical behaviours, while conforming to a rigorous method to maximize reusability and interoperability;
- modellers shall be properly educated on subjects such as the UML and the object-oriented paradigm – it is believed that these initial investments would lead to long-term payoff; and
- someone must be responsible for the integration in order to maximize the process efficiency.

Finally, the proposed M&S process only fosters model reusability and interoperability in providing guidelines to modeller teams. However, the object-oriented paradigm does not guarantee reusability and interoperability of the concepts. In order to achieve these requirements to a higher level, some constraints on the abstraction of entities, properties and interactions must be imposed [15].

6.0 REFERENCES

- [1] Fishwick, P.A., “*Simulation Model Design and Execution: Building Digital Worlds*”, Prentice Hall, New Jersey, 1995.

A M&S Process to Achieve Reusability and Interoperability

- [2] “*Creating and Improving Intellectual and Technological Infrastructure for M&S*”, Chapter 6, Technology for the United States Navy Corps, 2000-2035: Becoming a 21st-Century Force, National Academy of Sciences, 1997. (http://www.nap.edu/html/tech_21st/msindex.htm#Contents)
- [3] Booch, G., “*Object-Oriented Analysis and Design with Applications*”, The Benjamin/Cummings Publ. Comp. Inc., 1994.
- [4] The Unified Modeling Language (UML) Web Site: <http://www.uml.org>.
- [5] Gamma, E., Helm, R., Johnson, R. and Vlissides, J., “*Design Patterns: Elements of Reusable Object-Oriented Software*”, Addison-Wesley, 1995.
- [6] The RPR-FOM Standards Development Group Web Site: <http://www.sisostds.org/stdsdev/rpr-fom>.
- [7] Witkin, A. and Baraff, D., “*Physically Based Modeling: Principles and Practice*”, Course notes, SIGGRAPH, 1997.
- [8] Davis, P.K. and Zeigler B., “*Multi-Resolution Modeling and Integrated Families of Models*”, Appendix E, Chapter 6, Technology for the United States Navy Corps, 2000-2035: Becoming a 21st-Century Force, National Academy of Sciences, 1997.
- [9] The eXtensible Markup Language (XML) Web Site: <http://www.w3.org/XML>.
- [10] The HLA Web Site: <https://www.dmsomil/public/transition/hla>.
- [11] The Rational Rose Web Site: <http://www.rational.com/products/rose>.
- [12] The Xerces Open-Source XML Parser from Apache Web Site: <http://xml.apache.org>.
- [13] The MathWorks Web Site: <http://www.mathworks.com>.
- [14] The STRIVE Web Site: <http://www.cae.com/en/military/software/strive.shtml>.
- [15] Bernier, F., Poussart, D., Laurendeau, D. and Simoneau, M., “*Interaction-Centric Modelling for Interactive Virtual Worlds: the APIA Approach*”, Proceedings of the 16th ICPR conference, Quebec, Canada, 2002, pp. 1007-1010.





A M&S Process to Achieve Reusability and Interoperability

N. Harrison, B. Gilbert, M. Lauzon, A. Jeffrey,
C. Lalancette, R. Lestage and A. Morin

October 25th, 2002



Defence Research and
Development Canada

Recherche et développement
pour la défense Canada

Canada



Outline

- Problem
- Solution
- Proposed M&S process
- Suite of tools
- Incubating project
- Advantages
- Disadvantages
- Lessons learned
- Way ahead



1. Problem

Feature driven M&S



Non-reusable and non-interoperable M&S



Architecture driven M&S

How do we engineer models in order to build reusable, interoperable, extensible, modular and portable M&S applications?



2. Solution

Apply software engineering to M&S

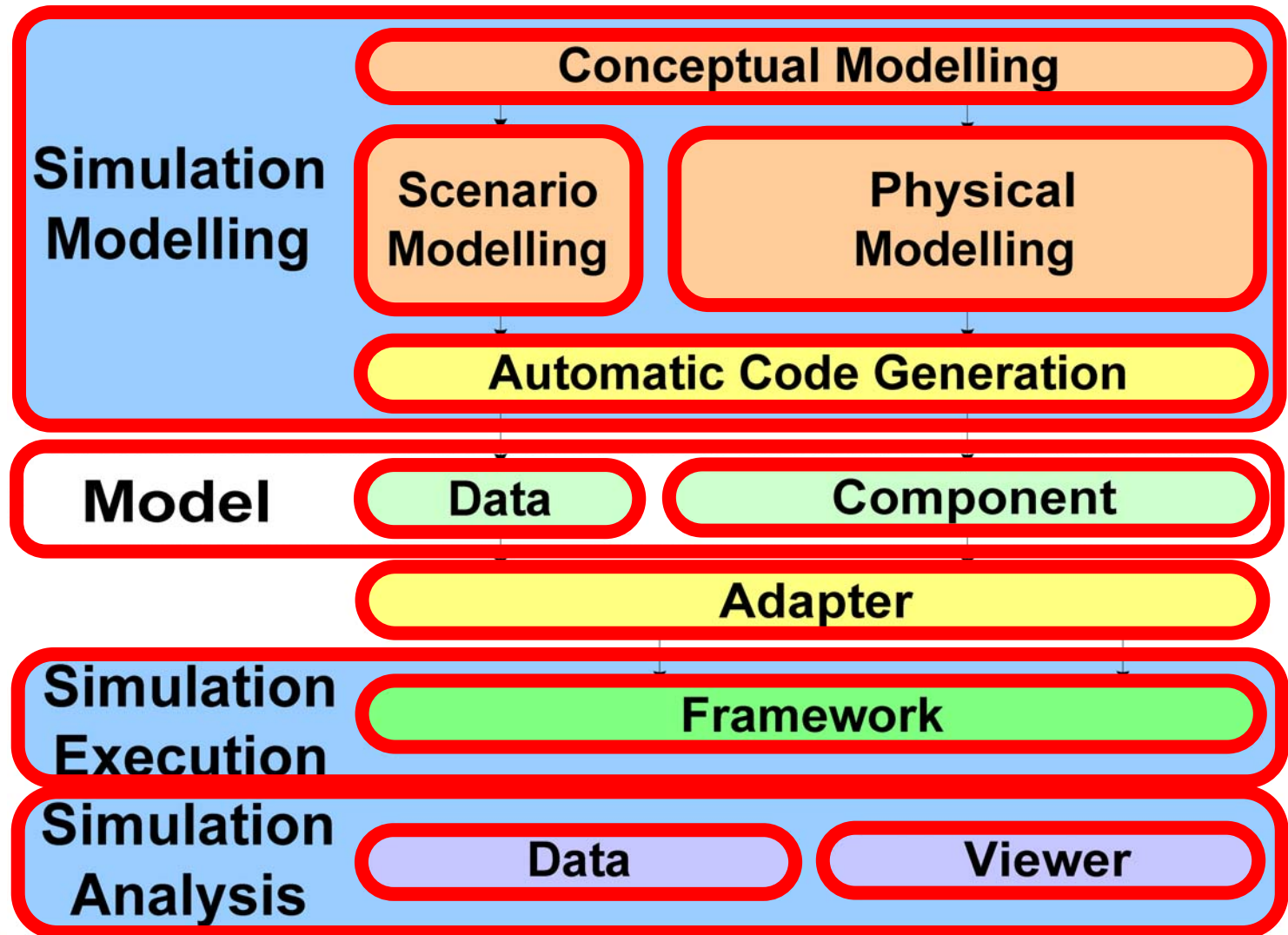


M&S Process

- Based on software engineering concepts, tools and best practices
- To guide the modellers
- Reusable and interoperable models
- Extensible simulation framework

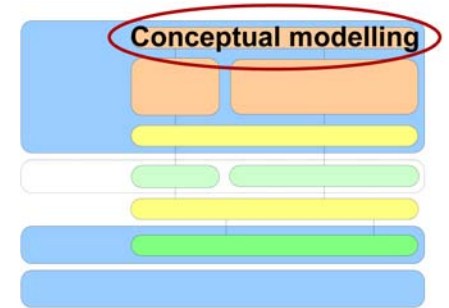


3. The M&S Process





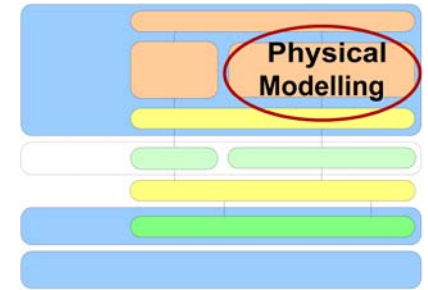
3.1 Conceptual Modelling



- Aim
 - Abstraction of the relationship between entities, properties, behaviours and interactions to be simulated
- Concepts
 - Object-oriented and component-oriented
 - Unified Modeling Language (UML)
 - Design Patterns and domain-specific standards
- Approach
 - Reference of design
- Benefits
 - Modularity, reusability, interoperability, extensibility



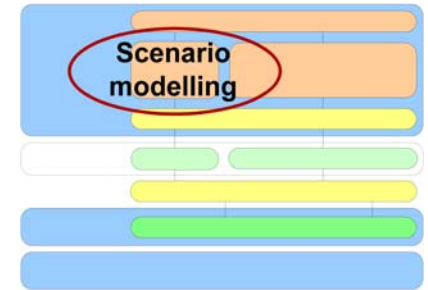
3.2 Physical Modelling



- Aim
 - Mathematical representation of entities, properties, behaviours and interactions to be simulated
- Concepts
 - Software programming
- Approach
 - Specialist modellers
 - Consistency with the conceptual model
 - Object-oriented programming or wrapping
 - Visual programming
- Benefits
 - Quality, consistency, interoperability, reusability



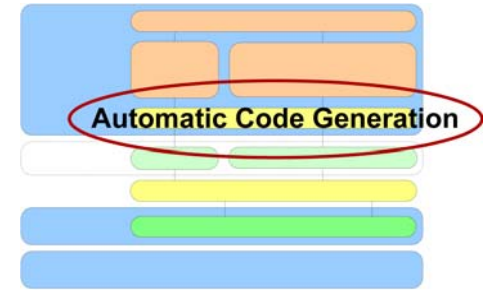
3.3 Scenario Modelling



- Aim
 - Configuration of simulation instances
- Concepts
 - Data representation standard
 - eXtensible Markup Language (XML)
- Approach
 - XML schemas
 - Parameters and initial conditions
 - Parts
 - Scenario
 - Log
- Benefits
 - Modularity, reusability, portability, dynamism



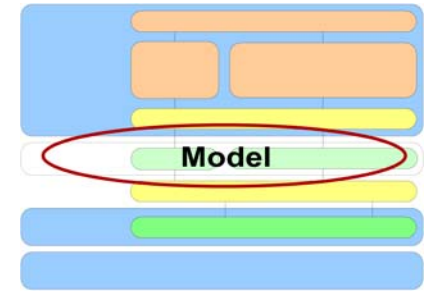
3.4 Code Generation



- Aim
 - Automatic software representation for the model components and data
- Concepts
 - Software integration and automation
- Approach
 - Speed up and standardize the end product
- Benefits
 - Quality, consistency, uniformity



3.5 Model Component / Data

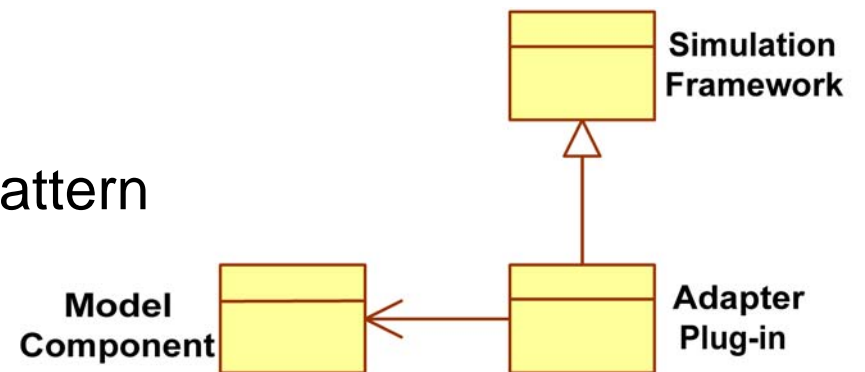
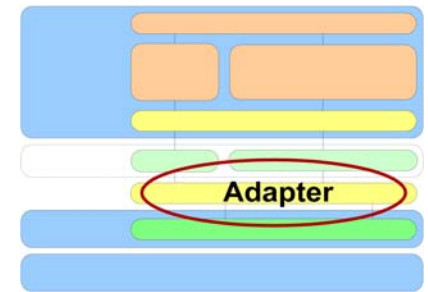


- Aim
 - Outcome of the modelling phase
- Concepts
 - Model = Component + Data
 - Component = generic object, dynamic, DLL
 - Data = specific instance, configuration, XML
- Approach
 - Pure model independent of the simulation framework
- Benefits
 - Modularity, reusability, dynamism



3.6 Adapter

- Aim
 - Adapt pure models to specific simulation frameworks
- Concepts
 - “Adapter” design pattern
 - Framework API
- Approach
 - Run-time selection of the model to be instantiated
 - Component plug-in
 - Scenario data import
- Benefits
 - Modularity, reusability, extensibility





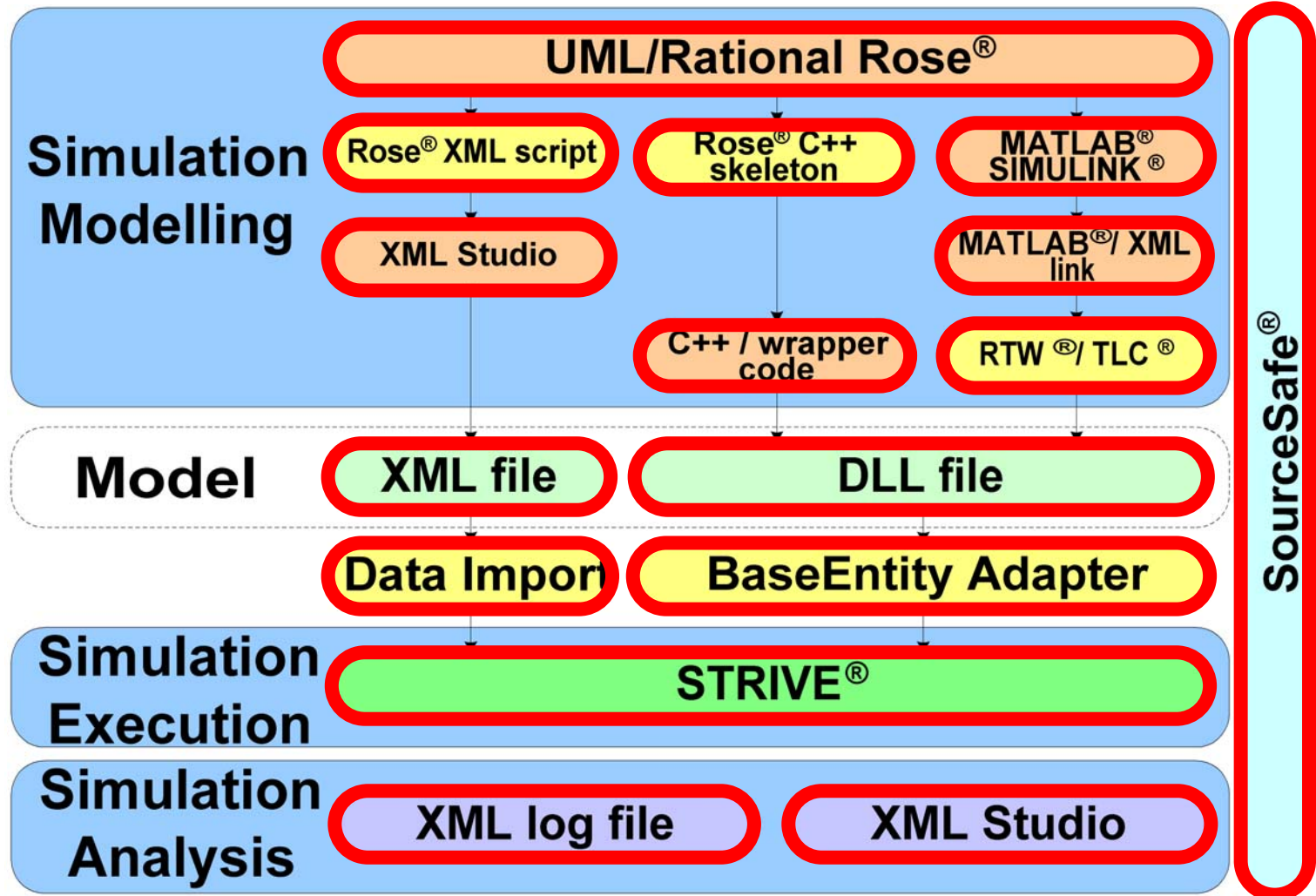
3.7 Simulation Framework



- Aim
 - Simulation and time management are left to a simulation framework
- Concepts
 - Common infrastructure
 - Object-oriented and component-oriented framework
- Approach
 - Take advantage of built-in functionalities like execution control, scheduling, visual scenario and doctrine creation, HLA compliance, distribution, trajectory waypoints, 2D and 3D viewers, etc.
- Benefits
 - Reusability, interoperability, extensibility



4. The Suite of Tools





5. The Incubating Project

- R&D project for weapon engagement simulation
 - to connect and interchange dynamically configurable subpart models developed by several specialists
- Conceptual model
 - Entities, Parts, Theatre and Environment
 - Standard: RPR-FOM (BaseEntity, WorldLocation, etc.)
- Physical Models
 - MATLAB/SIMULINK® weapons library
- Simulation Execution
 - Dynamic instantiation of configurable entities in STRIVE®



6. Advantages

- Interoperability
 - Agreement at the conceptual level and common framework infrastructure
- Modularity
 - XML data separated from DLL components
- Reusability
 - Modular models independent of any simulation framework
- Extensibility
 - Upgrade the conceptual model and add functionalities to the simulation framework
- Portability
 - Simulation data in XML format
- Quality
 - Consistency and uniformity preserved by automated steps



7. Disadvantages

- Maintenance of custom tools
- Rigorous information management
- Being at the mercy of COTS tools
- Learning curve
- Significant integration work



8. Lessons Learned

- Change of mind
- Transparent and efficient information sharing, appropriate communication and documentation
- Agreement at the conceptual model level
- Modellers must do what they are the best at, while conforming to a rigorous method
- Training is an initial investment that leads to long-term payoff
- Someone must be “responsible” for the integration



9. Way Ahead

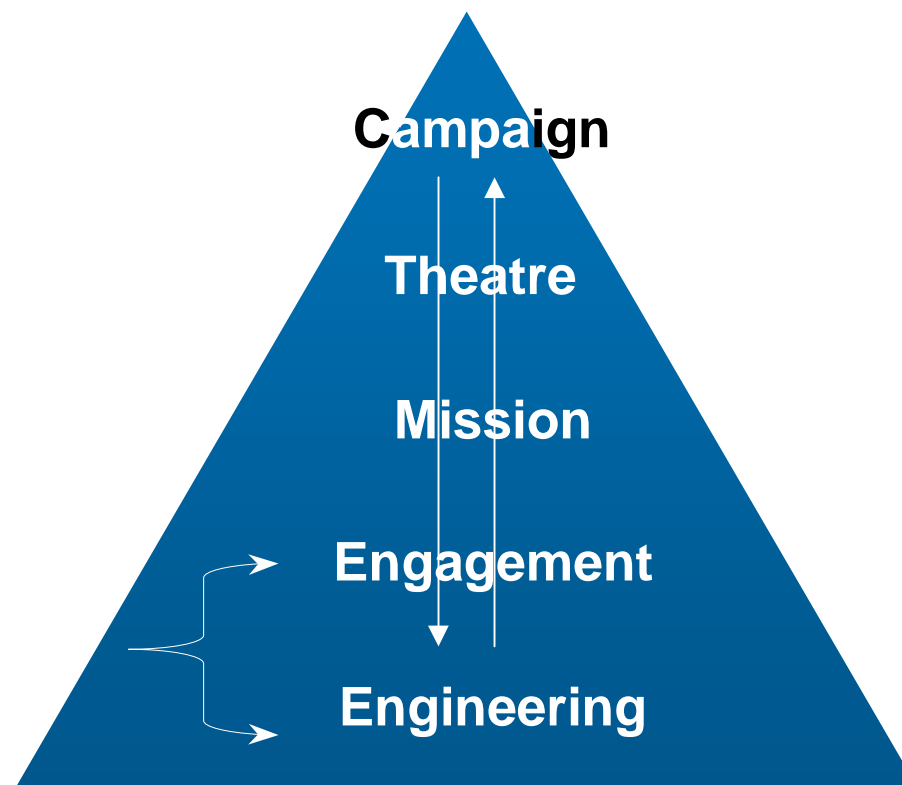
- From fostering ...
 - The proposed process only fosters reusability and interoperability by providing tools and guidelines to modellers
 - Object-oriented paradigm does not guarantee reusability and interoperability
- To achieving...
 - Constraints must be imposed on the conceptual abstraction of entities, properties and interactions
 - Meta-model to allow interaction between models without prior knowledge of each other

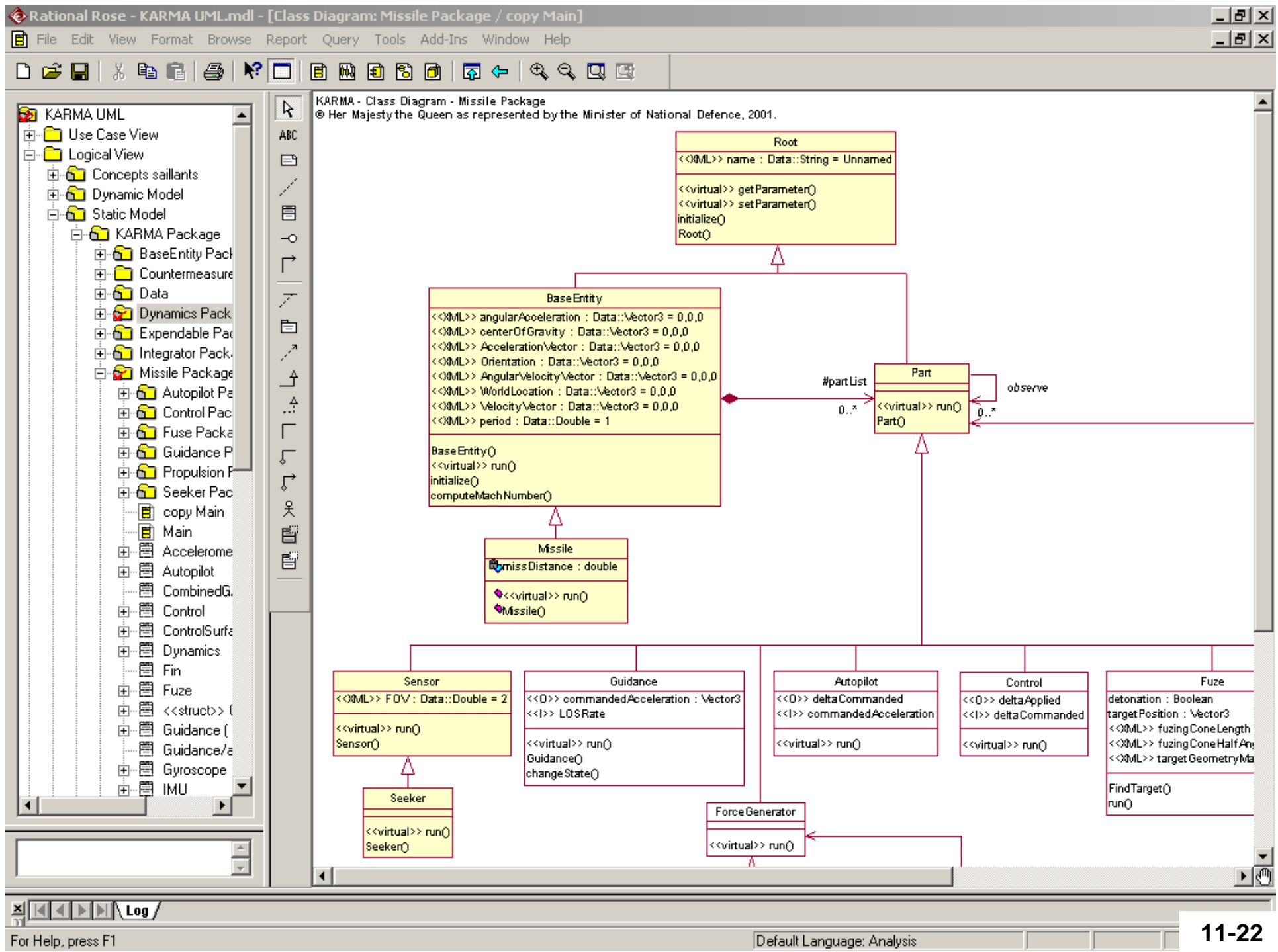
**Leader en sciences et
technologie de la défense,
la Direction de la recherche
et du développement pour
la défense contribue
à maintenir et à
accroître les compétences
du Canada dans
ce domaine.**





The M&S Pyramid





Rational Rose - KARMA UML.mdl - [Class Diagram: Missile Package / copy Main {read-only}]

File Edit View Format Browse Report Query Tools Add-Ins Window Help

KARMA - Class Diagram - Missile Package
© Her Majesty the Queen as represented

Root

```

<<XML>> name : Data::String = Unna...
<<virtual>>
<<virtual>>
initialize()
Root()

```

Class Specification for Missile {read-only}

Components	Nested	Files	C++	MSVC
General	Detail	Operations	Attributes	Relations

☒ Show inherited

Ster...	Name	Par...	Type	Initial
XML	name	Root	Data::SI Unname	
XML	angularAccel			
XML	centerOfGrav			
XML	Acceleration			
XML	Orientation			
XML	AngularVeloc			
XML	WorldLocatio			
XML	VelocityVect			
XML	period			
XML	missDistance			

Class Attribute Specification for Orientation {re...}

General Detail C++ MSVC

Name: Orientation Class: BaseEntity

Type: Data::Vector3 ☒ Show classes

Stereotype: XML

Initial value: 0,0,0

Export Control

☒ Public ☐ Protected ☐ Private ☐ Implementation

Documentation:

The yaw, pitch and roll angles between the entity's attitude and the reference coordinate system axes (calculated as the Tait-Bryan Euler angles specifying the successive rotations needed to transform from the world coordinate system to the entity coordinate system).

OK Cancel Apply Browse Help

BaseEntity()

```

<<virtual>> run()
initialize()
computeMachNumber()

```

Missile

```

missDistance : double
<<virtual>> run()
Missile()

```

Sensor

```

<<XML>> FOV : Data::Double

```

Guidance

```

<<Q>> commandedAcceleration : Ver

```

Autopilot

```

<<Q>> deltaCommanded

```

Control

```

<<Q>> deltaApplied

```

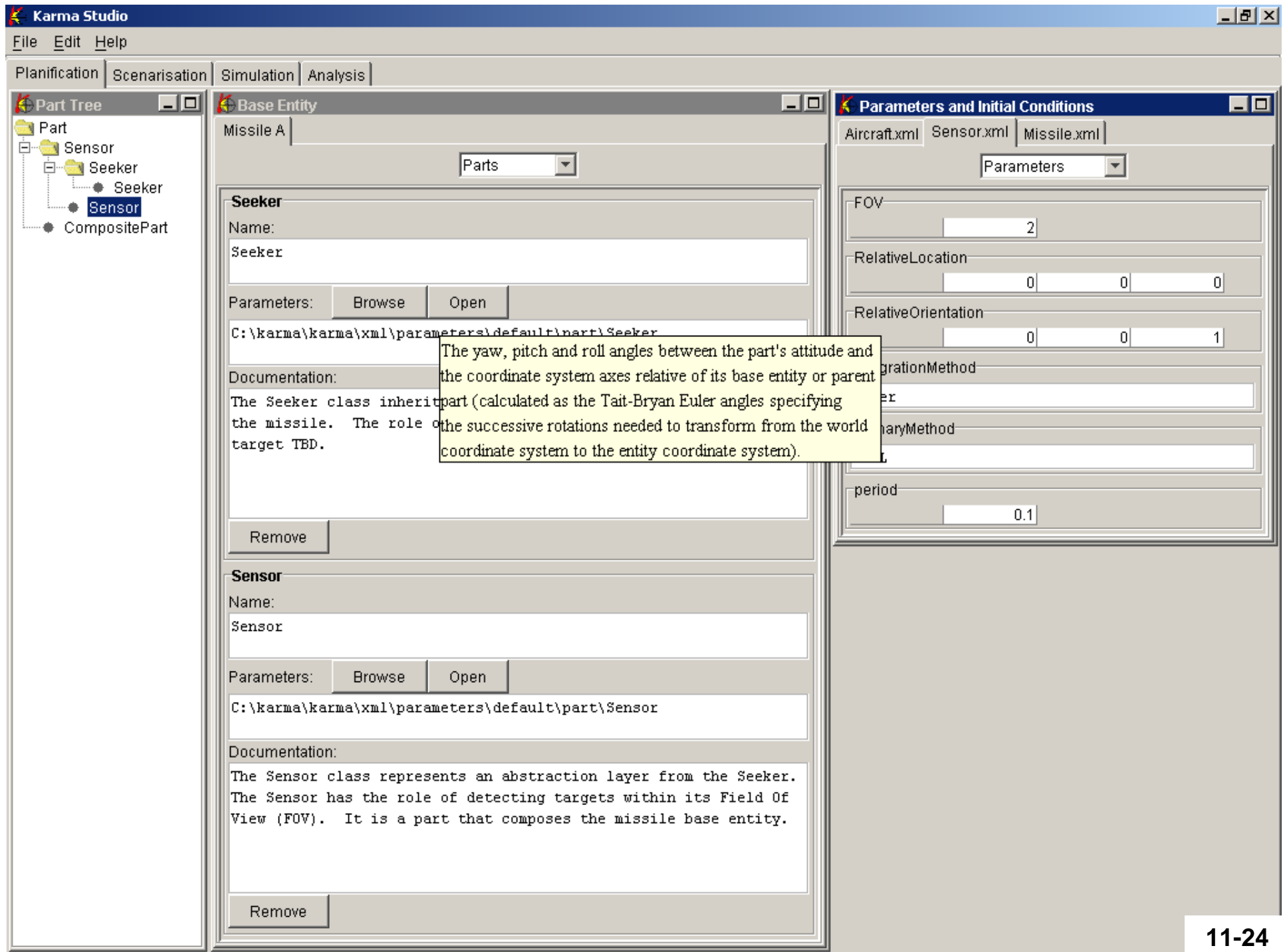
Fuz

```

detonation : Boolean

```

Default Language: Analysis



Rational Rose - KARMA UML.mdl - [Class Diagram: Missile Package / copy Main {read-only}]

File Edit View Format Browse Report Query Tools Add-Ins Window Help

KARMA - Class Diagram - Missile Package
© Her Majesty the Queen as represented by the Minister of National Defence, 2001.

Root

```

<<XML>> name : Data::S
<<virtual>> getParameter
<<virtual>> setParameter
initialize()
Root()

```

BaseEntity

```

<<XML>> angularAcceleration : Data::Vector3 = 0,0,0
<<XML>> centerOfGravity : Data::Vector3 = 0,0,0
<<XML>> AccelerationVector : Data::Vector3 = 0,0,0
<<XML>> Orientation : Data::Vector3 = 0,0,0
<<XML>> AngularVelocityVector : Data::Vector3 = 0,0,0
<<XML>> WorldLocation : Data::Vector3 = 0,0,0
<<XML>> VelocityVector : Data::Vector3 = 0,0,0
<<XML>> period : Data::Double = 1
BaseEntity()
<<virtual>> run()
initialize()
computeMachNumber()

```

Missile

```

missDistance : double
<<virtual>> run()
Missile()

```

Class Specification for Missile {read-only}

General Detail Operations Attributes Relations
Components Nested Files C++ MSVC

Set: default Edit Set...

Model Properties

Name	Value	Source
CodeName		Default
ImplementationType		Default
ClassKey	class __declspec(dll	Default
GenerateEmptyRegion	All	Default
PutBodiesInSpec	False	Default
GenerateDefaultConst	DeclareAndDefine	Default
DefaultConstructorVisi	Public	Default
InlineDefaultConstruct	False	Default
ExplicitDefaultConstru	False	Default
GenerateCopyConstru	DeclareAndDefine	Default
CopyConstructorVisibil	Public	Default
InlineCopyConstructor	False	Default
ExplicitCopyConstruct	False	Default
GenerateDestructor	True	Default
DestructedVisibility	Public	Default

Override Default Revert

OK Cancel Apply Browse Help

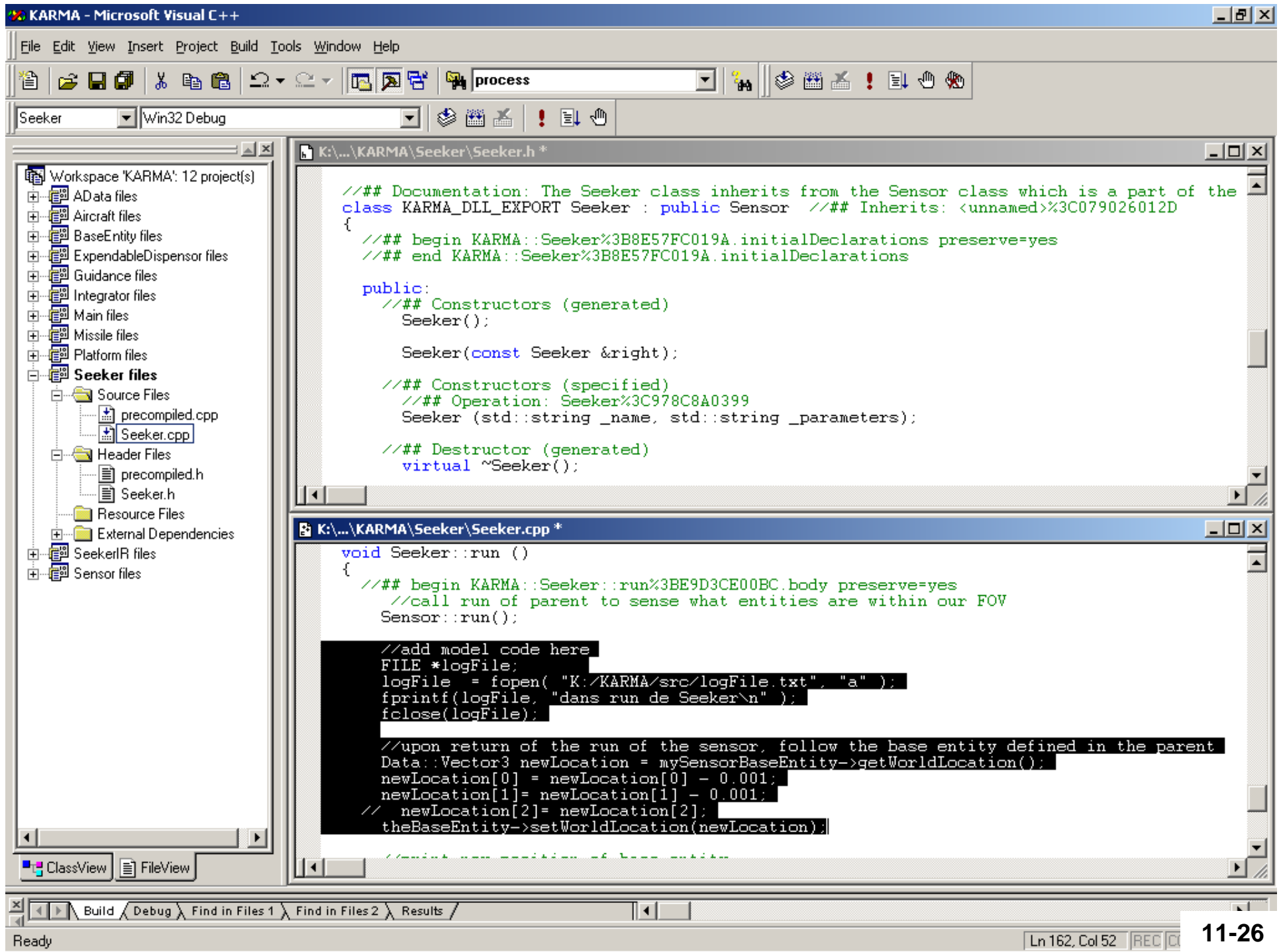
Open Specification...
Sub Diagrams
New Attribute
New Operation
Select In Browser
Relocate
Options
Format
C++
Edit

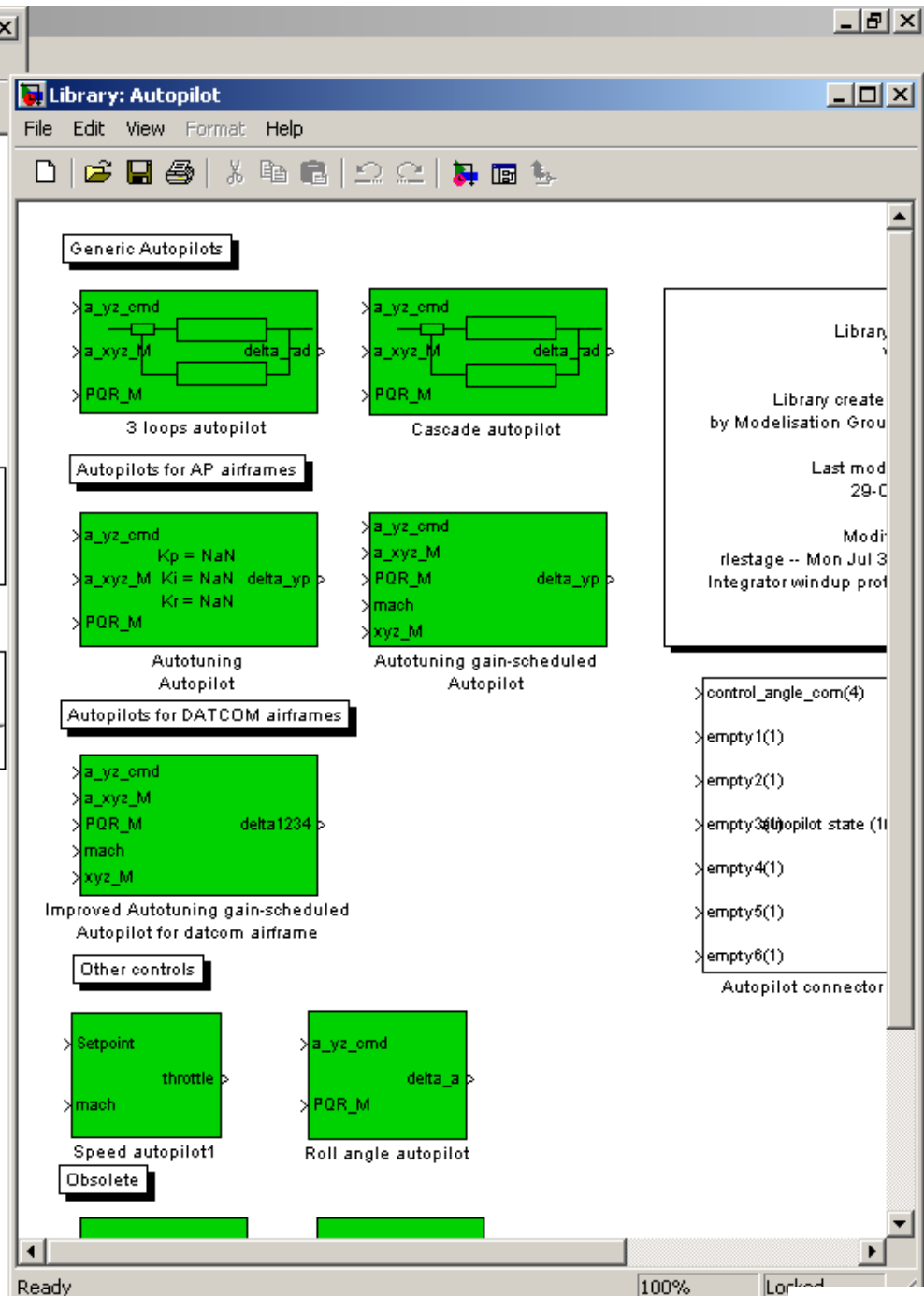
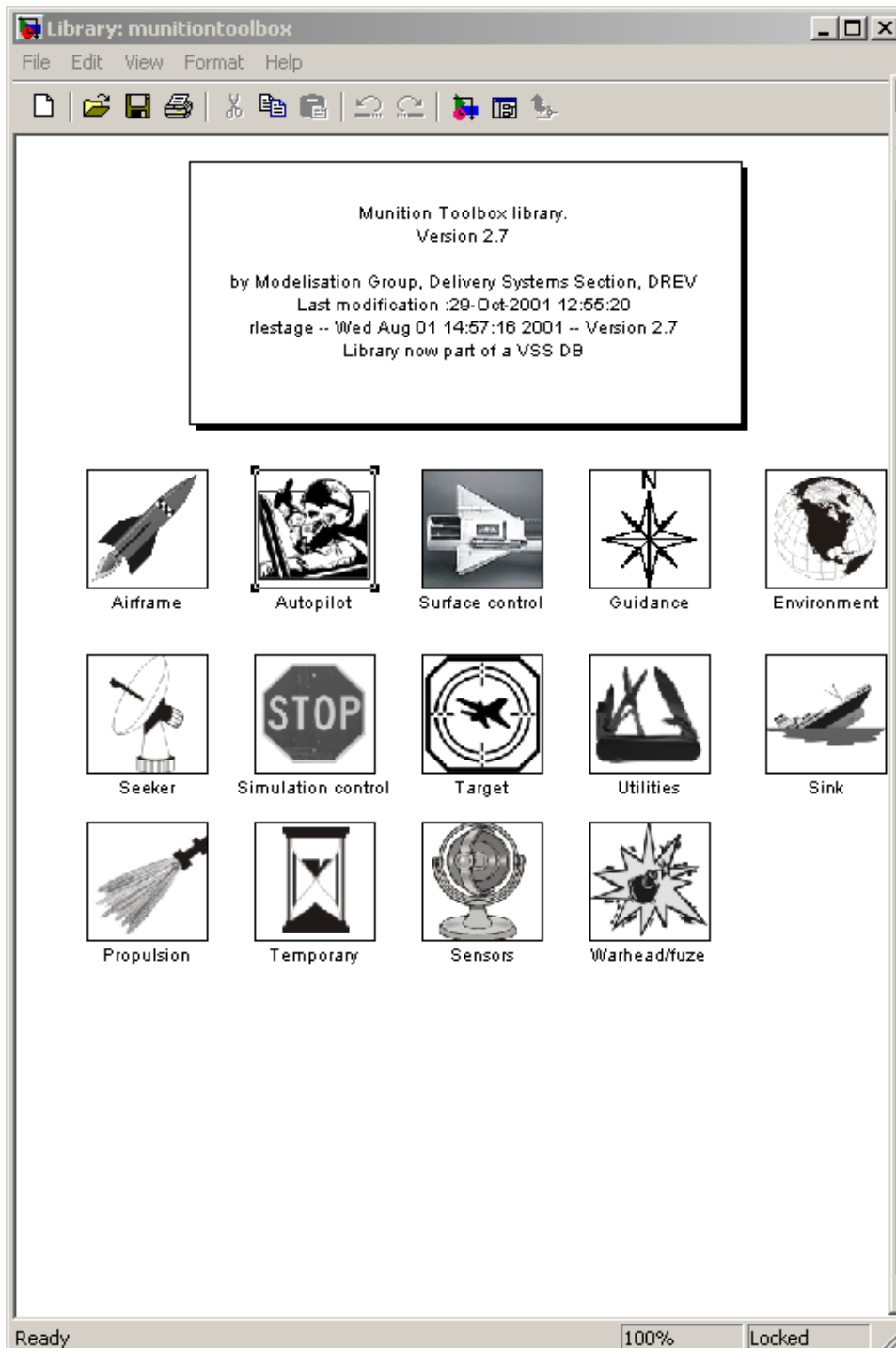
Code Generation
Browse Header
Browse Body

Log

For Help, press F1

Default Language: Analysis





Karma Studio for Matlab

File Edit Help

KarmaXMLToolbox *

File Edit View Simulation Format Tools Help

Normal

Karma XML Toolbox Block Library
Copyright 2002 Her Majesty the Queen as represented by the Minister of National Defence

Generate XML from a Simulink block

Configure a block with XML

Read XML into Matlab workspace

Ready 100%

roll angles between the entity's attitude coordinate system axes (calculated as the angles specifying the successive rotations from the world coordinate system to the entity)

Aircraft.xml

Initial Conditions

Orientation

WorldLocation

VelocityVector

CAircraft *

File Edit View Simulation Format Tools Help

Normal

AccelerationVector

WorldLocation

VelocityVector

Orientation

PQR_F

Aircraft

Ready 100%

ode4

Block Parameters: Aircraft

Bank to Turn Target (mask)

This block modelize a plane that bank at rate defined the roll speed vector and that applies an pitch acceleration of amplitude defined by the acceleration vector.

Parameters

Initial position [m]

WorldLocation

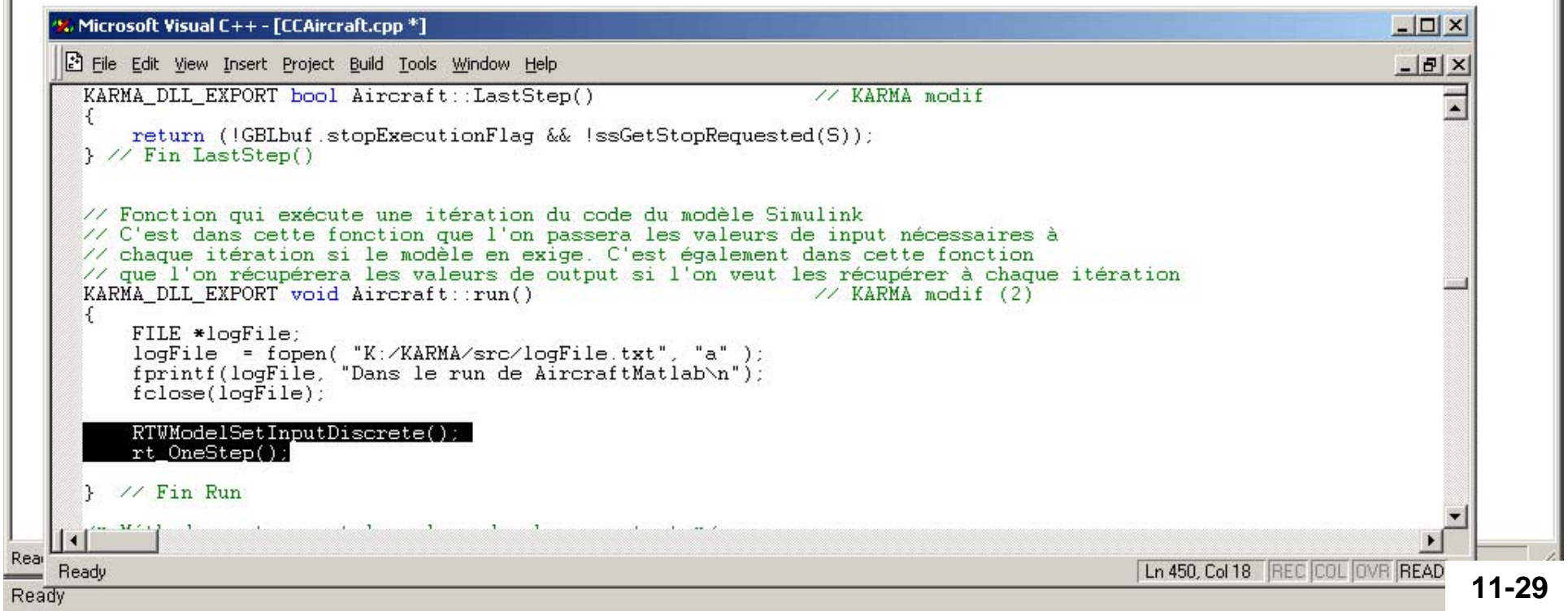
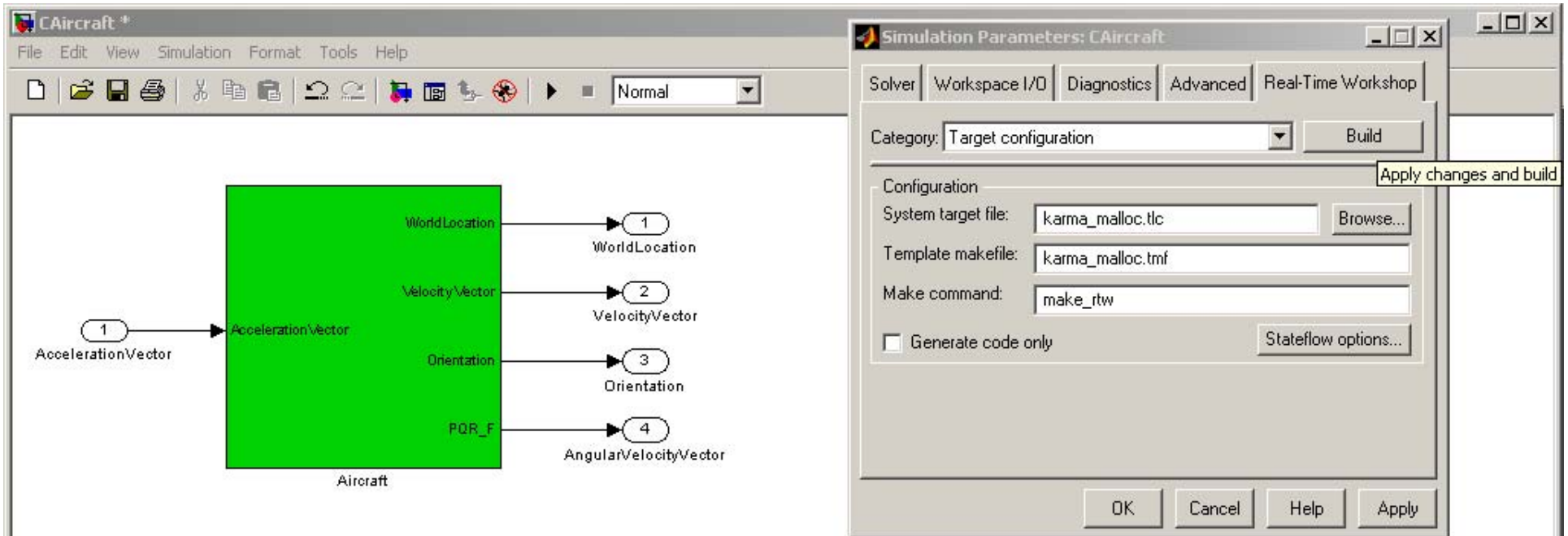
Initial velocity [m/s]

VelocityVector

Initial Euler angles [rad]

Orientation

OK Cancel Help Apply



Visual SourceSafe Explorer -- KARMA

File Edit View SourceSafe Tools Web Help

Working Folder: K:\KARMA\src\KARMA\MISSILE

Contents of \$/src/KARMA/Missile

Name	User	Date-Time	Check Out Folder
Missile.cpp	Nharriso	4/23/02 3:10p	K:\KARMA\src\KARMA\MISSILE
Missile.h		3/19/02 1:24p	
Missile.dsp		3/14/02 3:24p	
precompiled.h		3/10/02 10:43p	
precompiled.cpp		3/10/02 10:42p	

History of \$/src/KARMA/Missile/Missile.cpp

History: 12 items

Version	User	Date	Action
12	Jplouffe	3/19/02 1:25p	Checked in \$/src/KARMA/Missile
11	Jplouffe	3/19/02 1:07p	Checked in \$/src/KARMA/Missile
10	Nharriso	3/14/02 3:59p	Checked in \$/src/KARMA/Missile
9	Nharriso	3/14/02 11:09a	Checked in \$/src/KARMA/Missile
8	Jplouffe	3/01/02 9:34a	Checked in \$/src/KARMA/Missile
7	Nharriso	1/31/02 5:29p	Checked in \$/src/KARMA/Missile
6	Nharriso	1/30/02 5:30p	Checked in \$/src/KARMA/Missile
5	Nharriso	1/24/02 5:36p	Checked in \$/src/KARMA/Missile
4	Nharriso	12/05/01 4:52p	Checked in \$/src/KARMA/Missile
3	Nharriso	11/14/01 9:00a	Checked in \$/src/KARMA/Missile
2	Nharriso	11/09/01 4:34p	Checked in \$/src/KARMA/Missile
1	Nharriso	11/08/01 9:40a	Created

History Details

File: \$/.../Missile/Missile.cpp

Version: 12

Date: 3/19/02 1:25p

User: Jplouffe

Label:

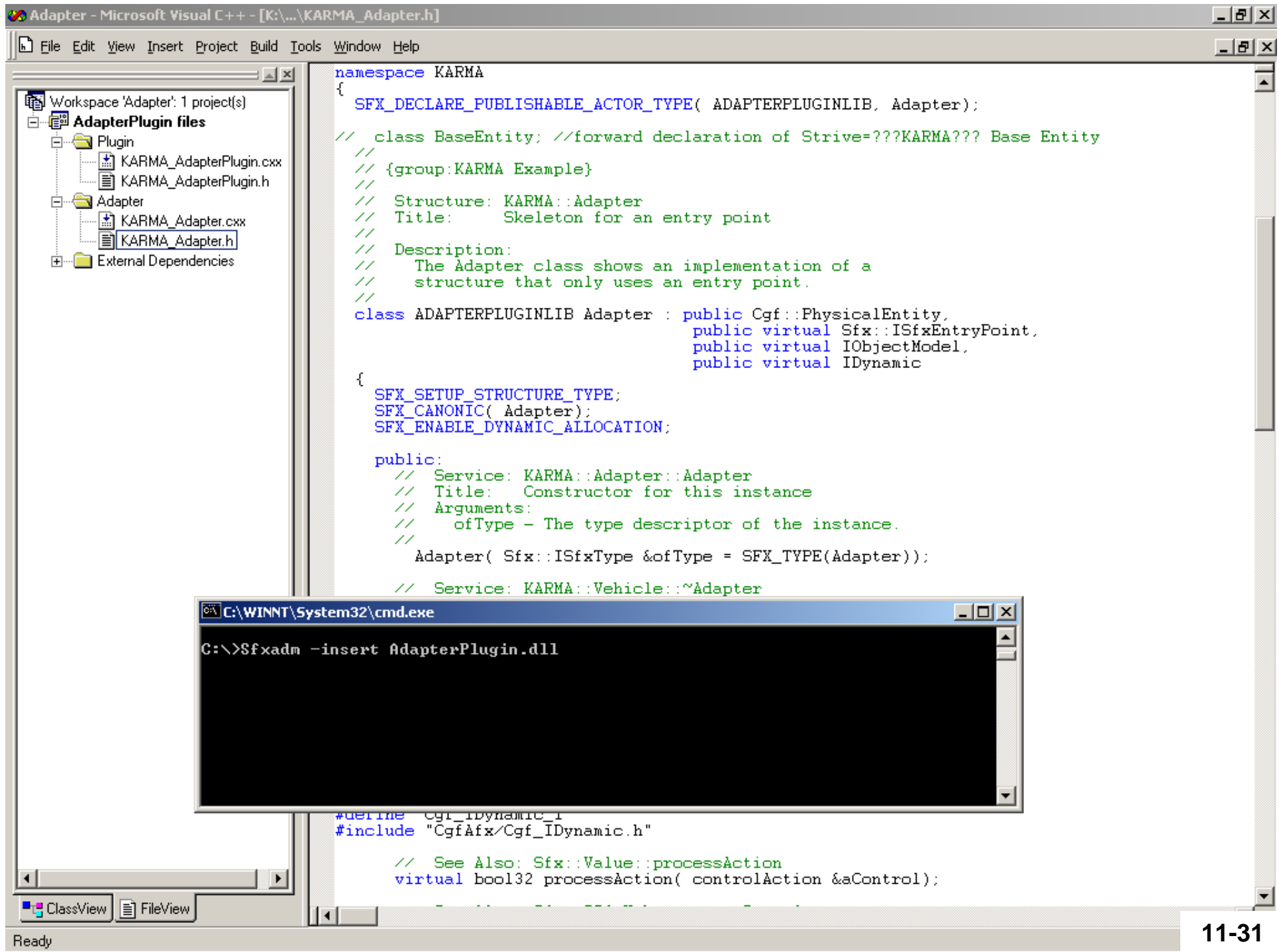
Checked in \$/src/KARMA/Missile

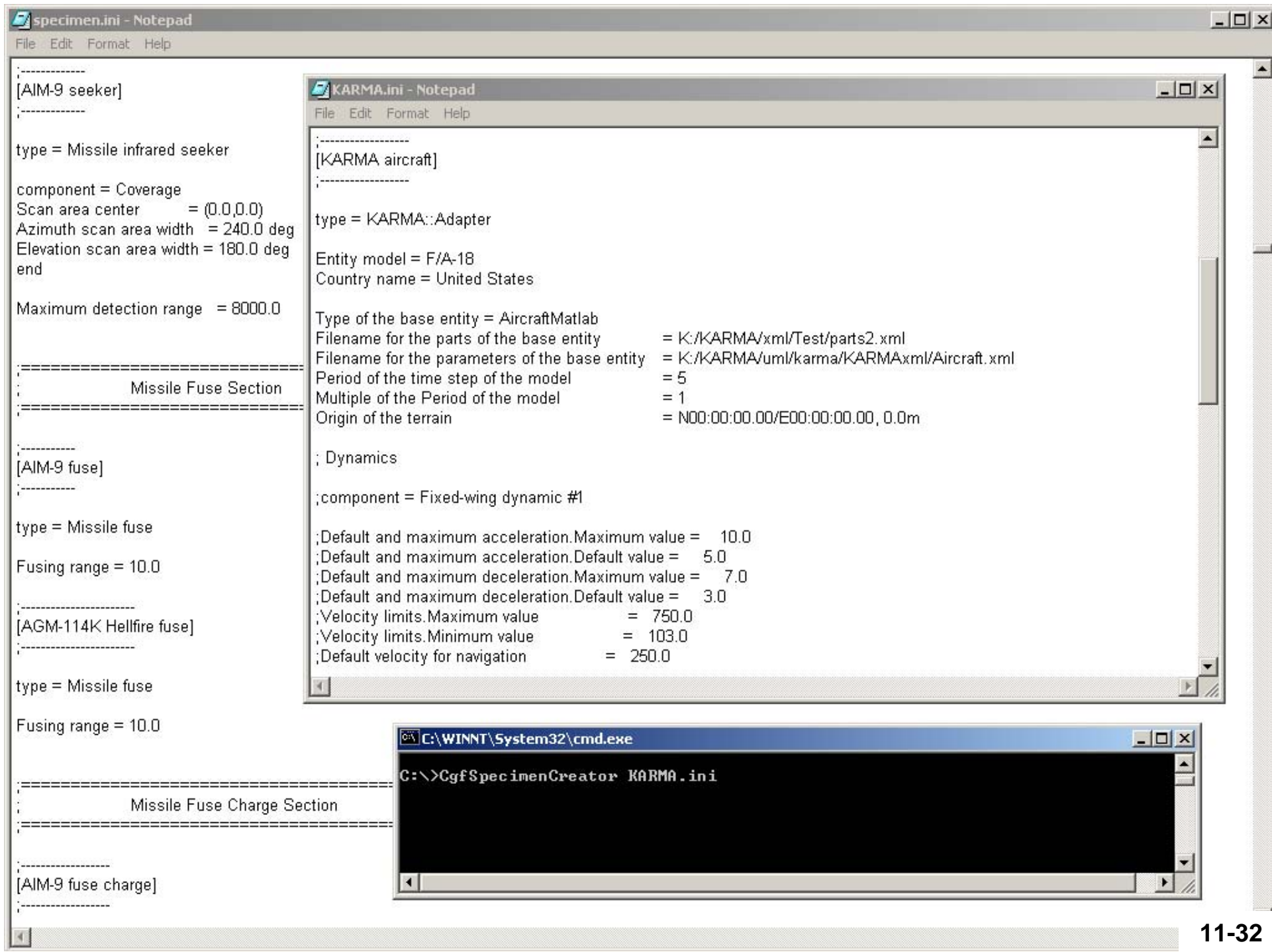
Comment: Code regeneration with correct model properties.

Label comment:

Ready

11-30





demo KARMA 2 - STRIVE Studio

File Edit Scenario Exercise View Help

Player: Scale: 1 : 39988

Air-Fixed wing

- A-10 attack aircraft model
- F-15 fighter aircraft model
- JSF ASTOVL fighter aircraft
- KARMA aircraft
- SU-27 Fighter

Air-Rotary wing

- Ground-Fixed
- Ground-Tracked
- Ground-Wheeled
- Surface-Ship
- Subsurface-Submarine

Other-Other

- KARMA missile
- Theatre

KARMA missile A

Description	Value
Damage State	No damage
Force identifier	Friendly
Identifies attached articulated parts	(null)
Type of the base entity	Missile
Filename for the parts of the base entity	K:/KARMA/xml/Test/partsMissileA.xml
Filename for the parameters of the base entity	K:/KARMA/xml/Test/paramMissileA.xml
Period of the time step of the model	1

Ok Cancel

KARMA missile B

Description	Value
Name	KARMA missile B
Damage State	No damage
Force identifier	Friendly
Identifies attached articulated parts	(null)
Type of the base entity	Missile
Filename for the parts of the base entity	K:/KARMA/xml/Test/partsMissileB.xml
Filename for the parameters of the base entity	K:/KARMA/xml/Test/paramMissileB.xml
Period of the time step of the model	1

Ok Cancel

Scenario Specimen

Log Query Status Instruments Rule Editor

Ready Editing N49:19:56.18/E08:49:15.44, 0.00m

11-33



Demo

